

Alma Mater Studiorum - Università di Bologna

**Dottorato di ricerca in
Lingue, Culture e Comunicazione Interculturale**

Ciclo XXI

Settore concorsuale di afferenza: 10/G1 - Glottologia e linguistica

Settore scientifico disciplinare: L-LIN/01 - Glottologia e linguistica

**Design, development and first evaluation of a
client/server system for managing and querying
linguistic corpora**

Tesi presentata da: Eros Zanchetta

Coordinatore dottorato

Prof. Marcello Soffritti

Relatore

Prof. Marcello Soffritti

Esame finale anno 2012

ABSTRACT

This thesis is concerned with the role played by software tools in the analysis and dissemination of linguistic corpora, as well as in their contribution to a more widespread adoption of corpora in different fields.

Chapter 1 contains an overview of some of the most relevant corpus analysis tools available today, presenting their most interesting features as well as some of their drawbacks. Chapter 2 begins with an explanation of the reasons why none of the currently available tools appear to satisfy the requirements of the user community and then continues with a somewhat detailed technical overview of the current status of the new system developed as part of this work. This presentation is followed by a few highlights of the features that make the system appealing to users and corpus builders (i.e. scholars willing to make their corpora available to the public). The chapter will conclude with an indication of future directions for the projects and information on the current availability of the software.

Chapter 3 describes the design of an experiment, carried out on a group of 58 students of translation, devised to evaluate the usability of the new system in comparison to another well-known corpus tool. Usage of the tool was tested in the context of a documentation task performed on a real assignment during an advanced translation class in a master's degree course.

In chapter 4 the findings of the experiment are presented on two complementary levels of analysis: firstly there is a discussion on how participants interacted with, and subsequently evaluated, the two corpus tools involved in terms of interface and interaction design, usability and perceived ease of use. Then an analysis follows of how users actually interacted with corpora to complete the task they were assigned and what kind of queries they submitted.

Finally, some general conclusions are drawn and areas for future work are outlined, both in terms of expansions and improvements of the current software platform and in terms of investigation on the behaviour of corpus tools users.

ACKNOWLEDGMENTS

First of all I would like to thank Professor Marcello Soffritti, my supervisor, for his support, advice and infinite patience in the years I spent working on this project.

Then a colossal thank you goes to Federico Gaspari, colleague, friend and spellchecker extraordinaire (this is the only section he has not read, so there will probably be some typos here) for his countless insightful comments and invaluable moral support.

The experiment (and consequently this thesis) would not have been possible without the input, collaboration and encouragement of my colleagues and friends Silvia Bernardini, Sara Castagnoli and Adriano Ferraresi. Thank you!

I also want to take this chance to thank Marco Baroni: he didn't have anything to do with this thesis, but he introduced me to the magic world of computational linguistics and for this I'll always be grateful.

I am also immensely grateful to the five heroes who braved the most brutal snowstorm in living memory to take part in the pilot study: I promised you anonymity and your secret is safe with me but I will not forget your names.

Finally, I would like to thank the 58 students who participated in the experiment. As for the 3 who never handed in their work at the end of the task and had to be excluded: it's OK, I forgive you.

TABLE OF CONTENTS

Abstract.....	I
Acknowledgments.....	III
Table of Contents.....	V
Table of Figures.....	IX
List of Tables.....	XI
Introduction.....	XIII
Chapter 1	
Corpus Analysis Tools.....	1
1.1 Available tools.....	2
1.1.1 Offline corpus tools.....	3
1.1.2 Online corpus tools.....	8
1.2 Corpora and copyright issues.....	20
Chapter 2	
Software Development.....	23
2.1 The requirements for a corpus manager.....	23
2.2 Limitations of the current approaches.....	25
2.3 Project Sarcophagus.....	29
2.3.1 Overview of the architecture.....	29
2.3.2 The Corpus Workbench (the back-end).....	30
2.3.3 Corpse (the middleware).....	34
2.3.4 Carcass (the front-end).....	35
2.3.5 How the system works.....	36
2.3.6 A more in-depth look.....	37
2.4 Advantages of the Sarcophagus system.....	39
2.4.1 Sarcophagus for users.....	40
2.4.2 Sarcophagus for corpus builders.....	45
2.5 Sarcophagus vs. Utopia.....	48
2.6 Future plans.....	50
2.6.1 Short-term developments.....	51
2.6.2 Medium-term plans.....	52
2.6.3 Long-term prospects.....	53
2.7 Availability.....	54

Chapter 3	
Evaluation Methodology.....	57
3.1 Experiment description.....	58
3.1.1 The original plan: a translation task.....	58
3.1.2 The actual plan: a documentation task.....	59
3.1.3 Resources.....	60
3.1.4 Test subjects.....	61
3.1.5 Pilot study.....	64
3.1.6 The experiment.....	68
Chapter 4	
Results and discussion.....	77
4.1 SUS score.....	77
4.2 Users' opinions.....	80
4.2.1 Feature comparison.....	80
4.2.2 Overall system comparison.....	83
4.2.3 Users intentions for future use.....	85
4.3 Analysis of users behaviour.....	86
4.3.1 Query modes.....	86
4.3.2 Query results.....	88
4.3.3 Student performance in the task.....	93
Conclusions.....	101
References.....	105
Appendix A.....	111
Appendix B.....	115
Appendix C.....	121
Appendix D.....	123
Appendix E.....	127
Appendix F.....	131
Appendix G.....	137
Appendix H.....	149

TABLE OF FIGURES

Figure 1.1: CucWeb's expert search.....	11
Figure 1.2: Korpus Südtirol.....	12
Figure 1.3: EPIC corpus in the SSLMIT Dev Online interface.....	14
Figure 1.4: word sketch of "landscape".....	18
Figure 2.1: basic architecture of Sarcophagus.....	29
Figure 2.2: main screen of Carcass.....	35
Figure 2.3: lifecycle of a typical user interaction (simplified version).....	36
Figure 2.4: Simple Query Editor.....	41
Figure 2.5: Expert Query Editor.....	42
Figure 2.6: Smart Query Editor.....	43
Figure 2.7: permissions.....	46
Figure 3.1: set-up of the experiment.....	70
Figure 4.1: mean and median SUS scores for Carcass and CQPWeb.....	78
Figure 4.2: distributions of the SUS score for Carcass and CQPWeb.....	78
Figure 4.3: "favourite" system as resulting from the comparison between the SUS score assigned by each participant.....	79
Figure 4.4: comparison of a subset of features of Carcass and CQPWeb.....	81
Figure 4.5: users opinion on which system they preferred and which one they thought was easier to use.....	84
Figure 4.6: breakdown of queries according to query mode.....	86
Figure 4.7: query results.....	88
Figure 4.8: distribution of errors according to query mode.....	89
Figure 4.9: percentage of 0-hits queries according to query mode.....	91
Figure 4.10: task grades.....	94
Figure 4.11: task score and second-year exam grade.....	95
Figure 4.12: worst five vs. best five scoring students.....	96

LIST OF TABLES

Table 3.1: education.....	63
Table 3.2: native language of the participants.....	64
Table 4.1: users opinions on what made one system better than the other.....	82

INTRODUCTION

This thesis is concerned with the role played by software tools in the analysis and dissemination of linguistic corpora, as well as in their contribution to a more widespread adoption of corpora in different fields. Acknowledging the limitations of currently available corpus management tools, the work describes the development of a novel system that allows a range of users, including those with limited technical expertise such as linguistics and translation students as well as scholars and researchers, to effectively consult corpora.

Background

The history of corpus linguistics is inextricably tied not only to the availability of data, but also to the tools that make it possible to analyse those data. In the early days of corpus linguistics, only researchers who had access to computer programmers and very expensive hardware could use corpora, since the tools to collect and analyse them were developed ad hoc for a particular dataset and often could only run on a specific machine (now we are used to software programs that run without problems on any computers, even on different operating systems, but this has not always been the case).

When, in the late 1980s, computers started to become cheaper, more compatible with each other and much easier to use, the software industry flourished: nowadays thousands of new computer programs

are developed every year to perform all kinds of tasks. Widely used applications like office productivity suites or graphic editing programs have reached incredible levels of sophistication; even open-source programs that can be downloaded for free like Firefox or OpenOffice have an exceptionally high quality in terms of performance, usability and features; this is possible because there are large teams of professional developers behind them.

Lamentably the same is not true for corpus analysis tools: even though the number of available programs in this area has seen a marked increase in the last two decades, there has not been the exponential increase in the quantity and quality of the tools seen in other fields. This is mainly due to economic reasons: the user base for corpus tools is formed primarily by a relatively small group of academics, who in many cases still prefer to develop their own tools rather than adopt ready-made solutions. It is therefore quite understandable that there would not be a great interest for software companies to invest in the development of programs that could only return rather slim profit margins. This is the reason why most of the existing commercial software solutions in the field of corpus linguistics (including those presented in chapter 1) were developed either by single individuals or small companies who have strong ties to the academic world. The same is not true for more profitable fields of linguistics, such as translation, which have a greater appeal for software houses that see a business opportunity in them (computer assisted translation software, for instance, are increasingly popular and companies like SDL and Kilgray invest in their development).

Similar considerations can also be made for open-source corpus tools: there are a few communities devoted to the development of free software, but they are very small and rarely (if ever) employ full-time programmers, relying instead on work done occasionally by students or researchers. This happens very often in open-source projects (especially

in academia), so much so that large source code repositories like Google Code report the level of activity of hosted projects as an indication of the appeal of the project (i.e. if a program has low activity then it is most likely new and not very stable or usable, or its development may have stopped altogether).

Statement of the problem

All these factors contribute to the fragmentation we see today: there are precious few tools which, on the whole, tend not to be as refined and usable as mainstream software programs, given the fact that they are mainly the result of projects developed by universities for internal use (possibly for work on a specific set of data) and then adapted for more general use. This kind of situation is hardly surprising given that what we are considering here are highly specialised tools used by scientists and not by the general public; in fact this is a common occurrence also in other disciplines. What makes the case of corpus linguistics unlike other fields however, like for instance physics or statistics, is that linguists, as a community, tend to be much less comfortable using computers than physicists and statisticians.

A number of problems derive from this situation: firstly, advanced use of corpora is confined to a few specialists in the field and is thus precluded to students, junior researchers or, more generally, linguists with limited computer skills. A second set of problem is represented by the difficulty in exchanging data, an impediment that makes it very hard to replicate results and verify studies made using different setups.

This is why an often made requirement for corpus tools is that they be more user-friendly (Hoffmann & Evert 2006), as many areas of linguistic investigation can benefit from the use of corpora, but all too often researchers lack the technical skills required to use them effectively. Moreover, corpora can also be used effectively outside academia, in areas such as language teaching and learning (Boulton

2010) and professional translation (Bernardini & Castagnoli 2008), where high levels of computer literacy are not very common.

Objectives of this work

My objective for this thesis was reviewing the state of the art in the field of corpus tools, analysing the main advantages and drawbacks of existing applications and trying to determine where current tools could be improved and how.

After that, I embarked upon the task of developing a new system that would be the foundation of an architecture capable of satisfying the needs of users (be they linguists, language teachers or translators) who do not have the technical skills – or the time and patience to acquire them – needed to use corpora effectively. Most importantly what I hoped to accomplish was the creation of a well-documented, appealing open-source project that could attract contributions from other like-minded individuals interested in bringing corpora to a wider audience.

An additional advantage offered by an easy to use, accessible platform that truly works with minimal effort, is that more people will be willing not only to use it, but also to share their resources through it.

Overview of the thesis

After this initial introduction, I will give an overview of some of the most relevant corpus analysis tools available today, presenting their most interesting features as well as some of their drawbacks. In chapter 2 I will explain the reasons why none of the currently available tools appear to satisfy the requirements of the community and then I will present, in some detail, a technical overview of the current status of the new system I developed as part of this work, accompanied by a few highlights of the features that make the system appealing to a wide variety of users and corpus builders (i.e. scholars willing to make their corpora available to the public). The chapter will conclude with an

indication of future directions for the projects and information on the availability of the software.

In chapter 3 I will describe the set-up of an experiment, carried out on a group of 58 post-graduate students of translation, devised to test the usability of the new system in comparison to another well-known corpus tool. Usage of the tool was tested in the context of a documentation task performed on a real assignment during an advanced translation class in a master's degree course.

In chapter 4 I will present the findings of the experiment offering two complementary levels of analysis: firstly I will discuss how participants interacted with, and subsequently evaluated, the two corpus tools involved in terms of interface and interaction design, usability and perceived ease of use. Then I will analyse how users actually interacted with corpora to complete the task they were assigned and what kind of queries they submitted; I will then formulate some hypotheses as to why they submitted certain types of queries and I will try to evaluate their degree of success.

Finally I will draw some general conclusions and outline areas for future work, both in terms of expansions and improvements of the current software platform and in terms of investigation on the behaviour of corpus tools users.

CHAPTER 1

CORPUS ANALYSIS TOOLS

The importance of corpus analysis software in the field of corpus linguistics has been stressed many times and lists of desiderata for an ideal corpus tool appear regularly in the literature (see for instance Hoffmann & Evert 2006 and Smith et al. 2008). The announcement for a recent workshop on "New developments in software tools for corpus construction and analysis"¹ read:

Those who use corpora for research, for learning and teaching languages, and for reference in their daily work, are well aware of the crucial (and sometimes frustrating) role played by software. One can spend a long time designing and compiling the perfect corpus, and yet be unable to make the best of it unless appropriate software is available for searching the corpus, exploiting the annotation and displaying results in a user-friendly way.

I will start with an overview of the most relevant software solutions available today, proposing a distinction between offline and online corpus tools. Although my enquiry will focus on the software and how users may interact with it, in the process of describing the tools, I will also give a few details about the resources such tools were designed for: this is inevitable given the tight link that in many case exists between the program and the underlying data.

¹ <http://goo.gl/4l3H2> (May 2, 2012)

I conclude the chapter by tackling the problem of copyright, a rather important issue that is often at the core of many of the design decisions made when developing online corpus tools

1.1 Available tools

In this section I will present an overview of the most notable and currently maintained corpus tools. For a more comprehensive analysis of corpus tools including benchmarks, see Wiechmann & Fuhs (2006) and, for a historical perspective, McEnery & Hardie (2012: 37-43) which also includes programs that are no longer available or obsolete. A thorough comparison of corpus programs for the extraction of lexical bundles was carried out by Ari (2006).

For the purposes of this analysis, we can divide existing corpus analysis tools into two broad categories: offline and online. By "offline" I mean applications that run on a computer and do not require a network connection to work, typical examples of this kind of programs include traditional word processors like *Microsoft Word* and spreadsheets like *Excel*. By contrast "online" applications require an active network connection and, especially in the last decade, often require a web browser to work; typical examples of online programs include web applications such as *Google Docs* (a full web-based office productivity suite, complete with word processor, spreadsheet and presentation program), and *Facebook* (a very popular social networking service).²

In recent years, the wide availability of Internet connections has contributed greatly to blurring the line between offline and online applications, yet the difference between the two remains very important when it comes to corpus analysis tools: with offline programs users need either to create or to own the corpus data they want to analyse, which could be a problem when perspective corpus users do not have the

² It should be noted that this kind of services are often thought of simply as webpages or websites, but the complexity of the operations they permit is such that the term "web application" is more appropriate (cf. also <http://goo.gl/04IDq>, April 28, 2012).

know-how, the time or the resources to acquire such data; conversely, online tools typically already include the corpora, so there is no need to obtain the data: this solves one set of problems (mainly related to copyright issues, cf. 1.2) but creates another since users who do own a corpus cannot use the application to analyse it given that online corpus tools typically only allow the use of a predefined set of corpora (to my knowledge, the only exception to this rule is the Sketch Engine, cf. 2.3.2.1).

1.1.1 Offline corpus tools

In this section I will present the main offline corpus tools available today. This is not a complete list of all available applications, only the most relevant, in terms of the ones that seem to be most widely used in academia, will be included.

1.1.1.1 *AntConc*

AntConc is a cross-platform tool (i.e. it runs on Windows, Mac OS X and Linux) "designed specifically for learners in a classroom context" (Anthony 2005: 7); the program is distributed as a freeware, which means that it can be downloaded for free from the project's website.³

AntConc implements a concordancer that supports regular-expression searches and displays results using a standard KWIC view, concordance lines can then be sorted using a number of criteria. It is also possible to visualise the distribution of matches in the corpus thanks to the "Concordance plot" tool. Additionally, the program is capable of generating frequency lists for unigrams as well as n-grams and can also extract keywords and collocations.

The interface of the program is clean and in general it is very easy to use, the only drawback is that performance rapidly degrades as corpus size increases, this is in line with the findings of Wiechmann &

³ <http://goo.gl/S8VXK>

Fuhs (2006) who claim that AntConc was not able to find concordances for high-frequency words in a 10-million tokens subset of the British National Corpus.

1.1.1.2 Paraconc

Paraconc (Barlow 2002) is a rather unique tool in that it can be used to query parallel corpora. The program is available for purchase⁴ and only runs on Microsoft Windows. At the time of writing, the file date on the most recent demo version downloadable from the web site is December 3, 2004 while the most recent version of the user manual is dated 2003; this could mean that development of the software has stopped, but the program works on modern operating systems (I tested it on Windows 7, 64-bit edition) and it is still possible to acquire a license.

The program accepts two separate lists of ASCII text files (each one representing one of the two languages of the parallel corpus) as input and is capable of aligning them automatically or to recognise different types of alignment delimiters: new line characters, arbitrary tags (e.g. <seg>...</seg>) or special markers indicating the beginning and end of segments (e.g. <seg> ID="XXX">...</seg>). Part-of-speech tagged corpora are also supported by the application.

ParaConc integrates all the typical features of corpus tools: when looking for patterns the program returns a list of concordance lines in which collocations are automatically highlighted; it is also possible to extract frequency lists from both active corpora. In addition, thanks to its support for multilingual texts, it is also capable of extracting the most likely equivalent of the searched expressions in the target language.

1.1.1.3 Wordsmith Tools

Wordsmith Tools (Scott 2008) is a very advanced, commercial corpus analysis tool for Microsoft Windows published by Lexical Analysis

⁴ <http://paraconc.com/>

Software Ltd. and Oxford University Press.⁵ The software has been under active development since 1996, is updated fairly regularly with new features and bug fixes and, at the time of writing, has reached version 6 (even though the latest update has not been released publicly yet).

Wordsmith Tools includes the basic four functionalities other corpus tools have: it implements a concordancer, a frequency lists generator, a collocations tools and allows keywords extraction based on n-grams. In addition to these fundamentals, it includes a host of other features like the possibility to use media markers to link audio files to a text, this is particularly useful for corpora containing transcriptions, because this makes it possible to listen to the original recording. Such corpora are not very common but a few examples exist, like the TED500 corpus created by Aston & Rodi (2012) using transcriptions and recordings from the TED website,⁶ or the EPIC corpus (Russo et al. 2006) containing transcriptions of European Parliament speeches (cf. 1.1.2.1).

Another tool in the kit can be used to identify "minimal pairs" (i.e. words that are minimally different from each other) in order to spot possible typos or anagrams in a text. The toolkit also contains a "Web getter" which allows users to automatically download web pages to build corpora on the fly: this facility is not as sophisticated as the BootCaT toolkit for corpus creation (Baroni & Bernardini 2004) but it has the advantage of being integrated in the corpus manager. *Wordsmith Tools* also supports user-defined markup which can be used to represent the structure of the text (e.g. sentences, paragraphs, etc.) or the part of speech of individual words in the corpus.

Wordsmith Tools can only run in Windows, which is unfortunate considering the increasing popularity of other operating systems. Also

⁵ <http://www.lexically.net/>

⁶ <http://www.ted.com/>

according to Wiechmann & Fuhs (2006) version 4.0 of the program was unable to perform queries for medium frequency words on the full BNC corpus (100 million tokens), although it is possible that the situation has improved with more recent versions of the program.

1.1.1.4 The IMS Open Corpus Workbench

The *IMS Open Corpus Workbench* (Christ 1994)⁷ is a set of command-line tools (i.e. tools that do not have a graphical interface) originally developed in the early '90s at the Institut für Maschinelle Sprachverarbeitung of the University of Stuttgart. Until recently the name of the suite was just IMS Corpus Workbench, "Open" was added in 2010 when the project became open source. The tools continued to be developed over the years becoming very popular among advanced users who were comfortable using a command-line interface. The Corpus Workbench has an exceptionally good performance (when compared to the other offline tools we presented above) and can support very large corpora (up to 2 billion tokens).

One of the defining characteristic of *CWB* is the capability to annotate corpora on 3 different levels: *positional*, *structural* and *aligned*.

At the **positional** level, each *position* (i.e. every token) in the corpus can be annotated with as many attributes as one wants, so for instance it is possible to add information about part of speech and lemma to every word in the corpus. The information added at this level is referred to as *positional attributes*; part of speech and lemma are the more commonly used positional attributes, but *CWB* is not limited in any way as to the number of annotations that can be added at this level.

Using the **structural** level of annotation, it is possible to add markers (in the form of pseudo-XML tags) that allow the identification of sentence or text boundaries, but also of other types of text

⁷ Often called simply Corpus Workbench or CWB, <http://cwb.sourceforge.net/>.

segmentation like syntactic phrases for instance. Additionally, each *structural attribute* can be annotated with metadata, so for instance it is possible to create a <text> attribute that encloses a text in the corpus and then add an annotation to indicate the source of that text, something like

```
<text id="http://foo.it">
```

The final level of annotation is the one that allows to add **aligned** regions of text, this allows *CWB* to manage parallel aligned corpora: the translation of a text is put into the *align attribute*, this allows users to query for a pattern in a language and find concordances in two languages.

The rich annotation supported by *CWB* enables users to perform sophisticated queries involving lexical as well as grammatical patterns, for instance it is possible to look simply for a sequence of words like:

```
(a) [word="black"] [word="cat"]
```

or one might choose to look for a different pattern:

```
(b) [pos="JJ.*"]+ [lemma="cat"]
```

would return all instances of the lemma "cat" preceded by one or more adjectives ("local cats", "suspicious wild cat", etc.). Note how query (b) shows how regular expressions are used *over characters* on the part-of-speech tags but also demonstrates that they can be used *over tokens*, that is to represent that the first token can be present one or more times in the pattern.

The *CWB* component used for concordance searches is the *Corpus Query Processor* (or *CQP*) which also gave the name to the query language, commonly known as "CQP syntax" or "CQP language", queries (a) and (b) above are formulated using this language. The two acronyms *CWB* and *CQP* are often used interchangeably.

Another key component of *CWB* is *cwb-scan-corpus*, a memory-efficient tool capable of computing the frequency of arbitrary patterns of

positional attributes in very large corpora. These patterns can also be very simple, for instance to generate a frequency list of all tokens in the corpus using *cwb-scan-corpus*, we could use a query like this one

(c) `word+0`

and if we wanted to extract a frequency list of all lemmas in the corpus we would submit this query:

(d) `lemma+0`

But the rich annotation of CWB allows more sophisticated uses: for instance if we wanted a list of all NOUN NOUN compounds in a corpus we could issue this command:

(e) `pos+0=/NN.* / pos+1=/NN.* /`

The Corpus Workbench is a very powerful tool which, unlike the other offline tools presented above, is capable of efficiently managing corpora of hundreds of millions of tokens. Unfortunately its greatest limitation is the lack of a graphical user interface: the original front-end for the toolkit was *Xkwic* (Lee & Rayson 2000) but it only worked in a Unix environment (i.e. not on Windows), moreover further development of the interface was completely dropped about ten years ago when the main developer deemed it a "poor design choice", a "bug" (Evert 2008: 8) and a "monolithic dead-end" (ibid: 37). Since for a very long time there has been no GUI (now the official interface is *CQPWeb*, cf. 1.1.2.1), the core of the user base of the Corpus Workbench is formed by people accessing it through one of the dozens web interfaces that have been built upon it in the last decade.

1.1.2 Online corpus tools

When the Internet became a global phenomenon in the late 1990s, many Universities saw that as an opportunity to make their corpora available to the public over the Net. Unfortunately at the time there were no ready-made, standardised tools to achieve this goal so many academic institutions started developing web interfaces for their

internal corpus tools. As a result, a myriad different interfaces were created, each tailor-made to the specific corpus that the institution wanted to make available, much like had happened in the early days of corpus linguistics when each research institute developed an internal set of corpus tools that could only run on a local mainframe (McEnery & Hardie 2012). In this section I will present a list of the most notable online corpus tools available today, dividing them into two groups: applications based on the Corpus Workbench and tools based on other platforms.

1.1.2.1 Corpus tools based on the Corpus Workbench

In the last decade, dozens of online corpus tools have been developed using the *IMS Open Corpus Workbench* as a back-end, the reasons for this incredible success are easily explained: its complete set of features, high performance and availability under a free and open-source license make it an ideal candidate for anyone wishing to build an online tool. This is why I will start with an overview of some of the most interesting web tools based on this software application.

Considering the remarkable influence this corpus continues to have in the field of corpus linguistics almost two decades after its publication, it seems appropriate to begin with the a web-based corpus tool that supports the analysis of the British National Corpus (Burnard 2007b), *BNCWeb CQP-Edition* (Hoffmann et al. 2008)⁸. This tool is a re-implementation of the original *BNCWeb* (Lehmann et al. 2000) using the *IMS Open Corpus Workbench* (see above) as a back-end instead of the original *Sara server* (Aston & Burnard 1998).

Thanks to its reliance on *CWB*, *BNCWeb CQP-Edition* supports the usual lexical method of finding concordances (i.e. search by word or phrase) but also allows users to exploit lemma and grammatical annotation.

⁸ <http://bncweb.lancs.ac.uk/>

In addition to the features made available by core CWB tools, the application also allows for query results to be further analysed using built-in statistical tools: it is possible to obtain data like the frequency breakdown of matches according to text category, the distribution of concordances in the various sections of the corpus and to extract collocations. It is also possible to assign each concordance line to a custom category for further analysis.

The tool makes it very easy to partition the BNC into subcorpora, for instance users can query just the texts belonging to the "spoken" part, or extract keywords from a particular subset of the corpus using another subset as a reference corpus. Access to the application is free, the only requirement is filling out a registration form using a valid e-mail address.

BNCWeb CQP-Edition was designed specifically with the British National Corpus in mind and is not easily adaptable for use with other corpora because many of the design choices that guided the development of the tool depended on the structure of that corpus (Hoffmann & Evert 2006). This is the reason why the *CQPWeb* project (Hardie forthcoming) was started, to create a clone of that application capable of supporting any corpus compatible with the underlying back-end (once again *CWB*).

I think it is fair to say that the goal has been accomplished given that, at the time of writing, the system already supports any corpus (in fact it is the tool I used as a baseline in my evaluation, cf. 3.1.2) and implements all the features present in *BNCWeb*. For more information on *CQPWeb* see also 2.2.

The *Leeds collection of Internet Corpora* (Sharoff 2006) is an ad hoc tool designed to make available a set of corpora built at the Centre for Translation Studies of the University of Leeds using texts downloaded from the web. All corpora have been created using a

Select corpus

WebCat A: 15 million words

Expert Search

Note: depending on the complexity of your query, it may take up to 1-2 minutes to obtain the results

	Position 1	Position 2	Position 3
Word	<input type="checkbox"/> NEG <input type="checkbox"/> MULTIPLE <input type="text"/>	<input type="checkbox"/> NEG <input type="checkbox"/> MULTIPLE <input type="text"/>	<input type="checkbox"/> NEG <input type="checkbox"/> MULTIPLE <input type="text"/>
Lemma	<input type="checkbox"/> <input type="text"/>	<input type="checkbox"/> <input type="text"/>	<input type="checkbox"/> <input type="text"/>
POS	<input type="checkbox"/> <input type="checkbox"/> <input type="text"/>	<input type="checkbox"/> <input type="checkbox"/> <input type="text"/>	<input type="checkbox"/> <input type="checkbox"/> <input type="text"/>
Syntax	<input type="checkbox"/> <input type="checkbox"/> <input type="text"/>	<input type="checkbox"/> <input type="checkbox"/> <input type="text"/>	<input type="checkbox"/> <input type="checkbox"/> <input type="text"/>
One or more	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Optional	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Search

Figure 1.1: CucWeb's expert search

variation of the BootCaT method (Baroni & Bernardini 2004). At the time of writing, corpora in the following languages can be used freely on the project's home page:⁹ Arabic, Chinese, English, Finnish, French, German, Greek, Italian, Japanese, Polish, Portuguese, Russian and Spanish. Two search modes are available on the website: the default mode allows users to search for simple phrases, while the "CQP syntax" mode puts the full power of the CQP language in the hands of the user. In addition to the facilities offered by CWB, the application can also find collocations using a set of Perl libraries written specifically for this project.¹⁰

Another CWB based tool is *CucWeb*¹¹ ("Corpus d'Ús del Català a la Web"), a corpus of Catalan built at the Pompeu Fabra University

⁹ <http://corpus.leeds.ac.uk/internet.html>

¹⁰ <http://sourceforge.net/projects/csar/>

¹¹ <http://goo.gl/V8Qa1>

Erweiterte Suche ? Expertensuche ? Mein Profil ? ?

Hund

Wortform

Sucheinschränkung ?

1. Wortform oder Lemma:
☒ Wortform ☐ Lemma

2. Wortart (optional):
☐ Adjektiv ☐ Adverb ☐ Artikel ☐ Konjunktion ☐ Nomen
☐ Partikel ☐ Präposition ☐ Pronomen ☐ Verb ☐ Andere

3. Groß-/Kleinschreibung:
☒ Beachten ☐ Ignorieren

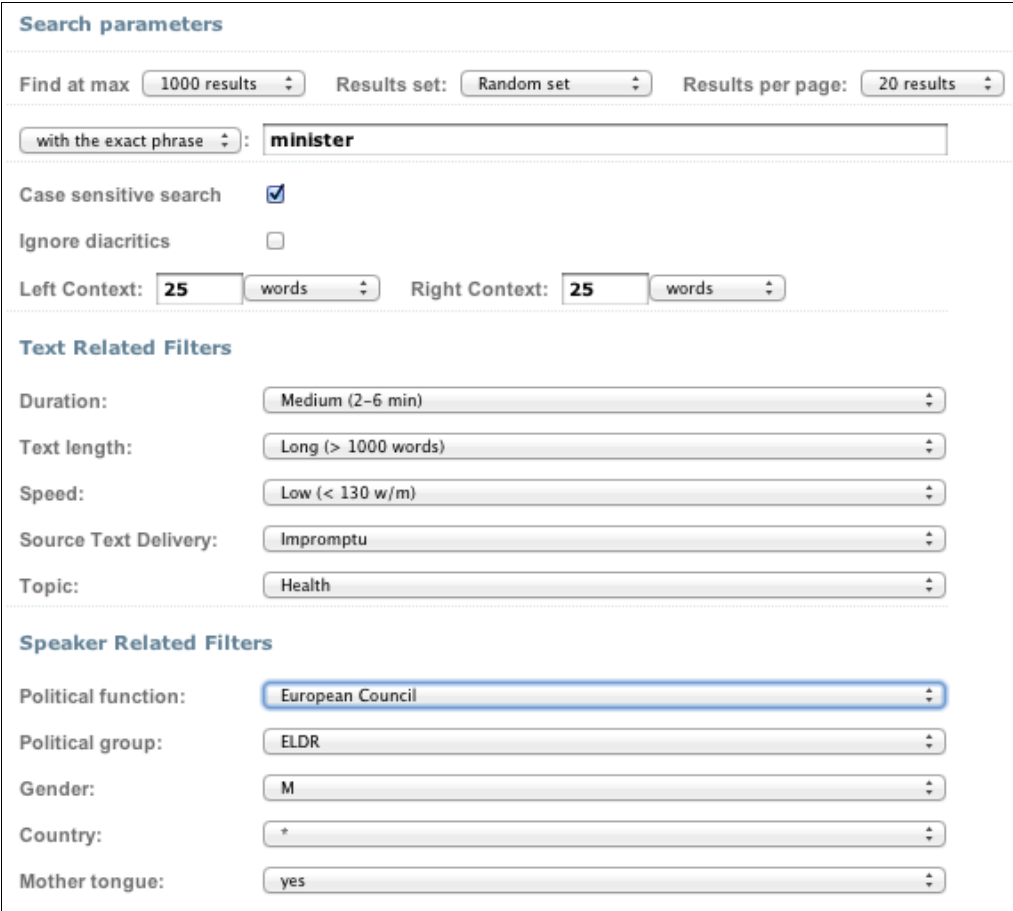
4. Anzahl:
 genau 1 oder von 1 bis 1

Figure 1.2: Korpus Südtirol

(Barcelona) using texts downloaded from the Internet (Boleda et al. 2004). This web tool presents a particularly interesting interface (see Figure 1.1) that allows users to compose sophisticated CQP queries employing a form that includes a series of checkboxes and drop down menus. This form served as an inspiration for the *Smart Query Editor* implemented by Carcass (cf. 2.4.1.3).

The beta version of the *Korpus Südtirol* (Anstein 2009)¹² web tool makes it possible to query four different corpora: 1) "Korpus Südtirol", a 2 million-token corpus of 20th century South Tyrol German); 2) "Dolomiten", a 66 million-token corpus built from texts obtained from the "Dolomiten" newspaper (Abel et al. 2009); 3) "DeWaC" and 4) "UkWaC", two large corpora of German and English respectively built by crawling the web (Baroni et al. 2009). Beside the usual "simple" and "expert" searches (which allow users to look for exact phrases or to use the CQP syntax respectively), this application includes a very interesting intermediate level called "advanced search" that offers the possibility to alter the search parameters on the fly though a pop-up window that

¹² <http://goo.gl/xqP92>



Search parameters

Find at max: 1000 results Results set: Random set Results per page: 20 results

with the exact phrase : **minister**

Case sensitive search: ☒ Ignore diacritics: ☐

Left Context: 25 words Right Context: 25 words

Text Related Filters

Duration: Medium (2-6 min) Text length: Long (> 1000 words) Speed: Low (< 130 w/m) Source Text Delivery: Impromptu Topic: Health

Speaker Related Filters

Political function: European Council Political group: ELDR Gender: M Country: * Mother tongue: yes

Figure 1.3: EPIC corpus in the SSLMIT Dev Online interface

appears by clicking on the tool-tips that show up right below the search box (see Figure 1.2).

The SSLMIT Dev Online¹³ website also offers access to multiple corpora: 1) "Repubblica", a 380-million-tokens corpus built collecting texts from the Italian newspaper "La Repubblica" (Baroni et al. 2004); 2) "EPIC", an open, parallel, trilingual (Italian, English and Spanish) corpus of European Parliament speeches and their corresponding interpretations (Russo et al. 2006) and 3) "Lorca", a 1-million word corpus of the complete works of Spanish poet and playwright Federico García Lorca (Piccioni 2008). This web application offers advanced facilities to exploit the rich structural annotations of the available resources: for newspapers corpora like "Repubblica" for instance it is

¹³ <http://dev.sslmit.unibo.it/>

possible to restrict the search by text author, genre, year of publication, etc.; likewise a spoken corpus like "EPIC" includes parameters such as delivery speed, duration, source text delivery, etc. (see Figure 1.3 for a screenshot of the query form of the EPIC corpus).

Another interesting set of corpora was developed by the University of Bologna's Interfaculty Centre for Theoretical and Applied Linguistics (CILTA) and made available on their website.¹⁴ The resources accessible through this tool include an 80 million-token corpus of contemporary texts chosen for their representativeness of the Italian language called "CORIS" and its companion corpus, "CODIS", designed to be easily partitioned into smaller sections in order to "create an extremely flexible corpus structure that can be adapted to almost any possible comparison with other reference corpora in different languages" (Rossini Favretti et al. 2001: 6). Other corpora available on the CILTA website are "BOLC", a multilingual corpus of legal texts in Italian and English (Favretti et al. 2001) and "DiaCORIS" a diachronic corpus of Italian texts produced between 1861 and 1945 (Onelli et al. 2006). Again, all these resources were made available through an interface built specifically for the project which uses the Corpus Workbench as back-end.

The bManuel site (Barbera 2007)¹⁵ also makes available a number of online corpora and offers an ample choice of interfaces: the website is somewhat chaotic and the same corpus can sometimes be accessed with up to three different (experimental) interfaces, but the service does offer free access to quite a few resources: "Corpus Taurinense", a historical corpus of 13th century Italian; "NUNC", a rather large set of multilingual newsgroup corpora and "Valico" a learner's corpus for students of Italian as a second language.

14 <http://corpora.dslo.unibo.it/>

15 <http://bmanuel.org/>

The Paisà corpus (Borghetti et al. forthcoming)¹⁶ is a large collection (about 250 million tokens) of Italian web texts licensed under a Creative Commons license, so unlike the vast majority of the other corpora we presented here, this one can also be freely downloaded from the web site (either in raw format or annotated in CoNLL format). This is probably also the reason why the online concordancer available on the website is rather rudimentary and appears to be just a simple demo to showcase the corpora.

This concludes the review of online corpus tools based on the Corpus Workbench. Many tools have been left out because they are not of particular interest or because they have been covered extensively elsewhere; readers interested in the subject may find more examples on the "Online Demos" page of the Corpus Workbench official website.¹⁷

1.1.2.2 Brigham Young University corpora

Brigham Young University (BYU) hosts a collection of seven corpora (Davies 2008):¹⁸ the *Corpus of Contemporary American English* (COCA, 425 million tokens), the *Corpus of Historical American English* (COHA, 400 million tokens), the *TIME Magazine Corpus of American English* (100 million tokens), the *BYU-BNC: British National Corpus* (100 million tokens), the *Google Book (American English) Corpus* (155 billion tokens), the *Corpus del Español* (100 million tokens) and the *Corpus do Português* (45 million tokens).

The website uses a proprietary technology based on a standard relational database (Microsoft SQL Server) for managing the corpora and implements, beside the usual searches, a set of rather unique features: using this custom search engine it is possible to integrate synonyms in the queries; for instance using this functionality, searching for "spectacular landscape" returns results like "breathtaking view" or

¹⁶ <http://www.corpusitaliano.it/>

¹⁷ <http://goo.gl/0s68Z>

¹⁸ <http://corpus.byu.edu/>

"dramatic setting". The application also offers a facility to compare words, which can be rather useful to appreciate the semantic and distributional differences or related concepts, for instance comparing "captive" and "prisoner" we can see that the first collocates with "breeding", "audience" and "animals" while the second collocates with "dilemma", "scandal" and "Azkaban". It is also possible to specify the part of speech of collocates and restrict the searches to subsets of the corpus (e.g. spoken, fiction, academic, etc.).¹⁹

The interface of the system is rather complex and it takes a little time to get used to it, although there is plenty of documentation on the website. The sheer number of available options can be daunting for the novice user and the fact that the various panels that make up the interface suddenly enlarge themselves when the mouse pointer moves over them can be off-putting at times. Finally, like all other online corpus tools, it is not possible to upload and use different corpora.

Despite all this, serious users will find that the system is very powerful and offers unmatched search capabilities. Moreover, access to the service is completely free: casual users can submit about a dozen queries before being asked to fill out a simple (and free) registration form.

1.1.2.3 Sketch Engine

The *Sketch Engine* (Kilgarriff et al. 2004)²⁰ is an online corpus tool that gives access to about 80 different corpora (some of which are very large, up to 20 billion tokens) in 42 languages, including Arabic, Chinese, Hebrew and Japanese, besides the usual selection of European languages.

¹⁹ All examples reported above were taken from COCA (which contains texts produced between 1990 and 2011), but all corpora on the website benefit from the same set of functionalities.

²⁰ <http://www.sketchengine.co.uk/>

landscape (noun) British National Corpus freq = 3829 (34.1 per million)

object of	637	1.6	subject of	237	1.1	modifier	1687	1.5	modifies	702	0.7	and/or	669	1.3
litter	8	7.42	inspire	5	5.16	lunar	11	7.52	gardener	28	8.43	seascape	5	7.77
dot	7	7.38	surround	9	4.87	barren	11	7.4	gardening	16	8.14	townscape	5	7.74
conserve	7	7.32	change	12	3.62	bleak	13	7.33	painter	27	7.71	still-life	4	7.43
scar	5	7.23	represent	5	2.56	rural	36	7.09	architect	34	7.69	wildlife	19	7.38
blot	4	7.08	remain	6	2.51	upland	11	7.04	painting	48	6.93	portrait	17	6.83
paint	19	6.49	become	13	2.22	wooded	8	6.96	conservation	9	6.1	amenity	5	6.39
survey	7	6.48	look	9	1.75	rugged	8	6.96	evolution	9	5.92	coastline	4	6.32
dominate	24	6.35	begin	5	1.52	surrounding	13	6.95	architecture	11	5.73	scenery	4	5.98
alter	12	5.83	seem	5	1.17	Stowe	7	6.94	photographer	6	5.7	monument	5	5.72
transform	9	5.67				rocky	9	6.91	designer	9	5.69	habitat	4	5.36
spoil	4	5.62	adj subject of	92	2.0	urban	27	6.77	evaluation	7	5.34	heritage	5	5.13
preserve	9	5.54	flat	5	5.32	valued	6	6.7	feature	32	5.3	climate	6	4.98
explore	7	4.95				english	56	6.65	design	27	5.07	conservation	4	4.94
view	6	4.71				desolate	6	6.64	drawing	8	4.98	interior	4	4.87
protect	10	4.56				romantic	11	6.57	garden	21	4.88	architecture	5	4.6
improve	9	4.21				arid	6	6.56	quality	26	4.74	painting	7	4.15
create	16	3.82				flat	16	6.52	historian	4	4.48	deposit	4	4.08
enjoy	8	3.63				rolling	8	6.5	artist	7	4.02	building	14	3.53
study	5	3.6				historic	11	6.44	change	30	3.84	picture	7	3.19
record	6	3.56				undulating	5	6.42	value	20	3.82	scene	4	3.13
affect	9	3.53				lowland	6	6.37	impact	5	3.56	animal	5	2.83
change	10	3.34				wild	18	6.36	assessment	5	3.5	village	4	2.6
recognise	4	3.19				beautiful	23	6.23	description	4	3.42	environment	5	2.58
produce	12	2.9				stunning	6	6.22	improvement	4	3.39	site	5	2.5
form	7	2.86				familiar	17	6.16	scene	4	3.13	garden	4	2.49

Figure 1.4: word sketch of "landscape".

The system is based on the open-source *Manatee* engine (Rychlý 2007, cf. 2.3.2.1) which implements CQL, an extension of the CQP language used by the *IMS Open Corpus Workbench* (cf. 1.1.1.4).

One of the most notable features of the system is its ability to produce "word sketches", that is "corpus-based summary of a word's grammatical and collocational behaviour".²¹ For instance the sketch of a word like "landscape" (see Figure 1.4) can be used to determine which words combine with it in a different grammatical relations (e.g. when it is the object of verbs like "litter" or "dot", when it is modified by adjectives like "lunar" or "rocky" or when it acts as a modifier of nouns like "painter" or "architect"). It is also possible to compare the sketches of two words: in this case the system will show the patterns that are common to both of them and those that are specific to each word.

²¹ <http://goo.gl/x9jl4> (December 5, 2011)

The *Sketch Engine* is also capable of automatically building thesauri from corpora, so users can look for words that are related to one another in terms of distribution (i.e. words are considered related when they occur in similar contexts, that is when they share a certain number of collocates).

Another unique feature of the system is the possibility of building corpora using the *WebBootCaT* tool, a re-implementation of the *BootCaT Tools* (Baroni & Bernardini 2004)²² for the fast construction of corpora from the web. Corpora created with this tool can then be used directly in the *Sketch Engine*. In addition to this, paying customers can also request the installation of custom corpora created using other methods.

The *Sketch Engine* implements a clean interface aimed primarily at advanced users: only a minimal effort was made to make it intuitive for neophytes since a certain degree of competence in the CQL is taken for granted. Thanks to the Manatee engine, the system is fast and responsive, even when working with very large corpora.

The *Sketch Engine* is a commercial service: users can register for free and use the system for up to 30 days, after which they are required to pay a subscription fee to continue using it.

1.1.2.4 Michigan Corpus of Academic Spoken English

MICASE, the "Michigan Corpus of Academic Spoken English" (Simpson et al. 2002),²³ is a collection of transcribed speech events created at the University of Michigan. The search functionalities offered by the concordancer are pretty basic: only a simple phrase search is available, with the option of restricting results according to parameters like speaker gender, academic position, native language, etc.

²² <http://bootcat.sslmit.unibo.it/>

²³ <http://goo.gl/pvApo>

The tool also offers the possibility to browse the text directly, without searching, and to listen to the original recording of transcribed texts, even though there is no direct mapping between the corpus data and the audio files (i.e. it is not possible to jump from a concordance line to the recording from which that line was transcribed). The corpus is rather large (considering the fact that it is a corpus of transcriptions) at about 1.8 million tokens and can be accessed freely (no registration is required).

1.1.2.5 *WebAsCorpus*

Projects like *Wacky*²⁴ propose the construction of large, annotated corpora by downloading, post-processing and annotating documents from the web. The corpora produced following this methodology are very similar to more traditional corpora like, for instance, the BNC.

By contrast, the *WebAsCorpus* (Fletcher 2007)²⁵ online tool adopts a different approach: when the user submits a query, the system connects directly²⁶ to the *Bing* search engine and finds URLs matching the user's request. *WebAsCorpus* then downloads Bing's cached copy of the pages and presents them to the user highlighting the searched phrase. Users can then browse the obtained results directly in *WebAsCorpus*, or they can decide to download all the pages found by the system as a single zip file. This is very convenient for downloading quickly a large set of pages and construct a small corpus.

The system is completely free, it supports more than 20 languages and offers an "advanced query" in which it is possible to specify additional search parameters. At the time of writing, the future of the system appears uncertain because of its reliance on the free Bing Search

²⁴ <http://wacky.sslmit.unibo.it/>

²⁵ <http://webascorpus.org/>

²⁶ The connection is accomplished using Bing's Application Programming Interface (or API), a special protocol that makes it possible for computer programs to communicate with one another without human interaction; in this case the two programs involved are the *WebAsCorpus* application and the Bing search engine.

API: in April 2012 Bing announced that version 1.0 of the API would be decommissioned and that it would stop working on August 1, 2012.²⁷ The new version of the API will allow only 5000 free queries a month for each account, therefore it seems unlikely that WebAsCorpus will be able to continue to work in its current form.

1.2 Corpora and copyright issues

One might wonder why so many different online tools, most of which free, have been created for analysing corpora, why not just make them available as text files on a website for people to download and analyse using an off-line corpus tool? The reason is that most of the texts contained in corpora are copyrighted and cannot be freely distributed.

While there are corpora that can be bought off the shelf like the British National Corpus (Burnard 2007b) or "Le Monde",²⁸ the vast majority of corpora that have been developed in the past decade, especially in academic institutions, were built using text downloaded from the Internet, examples of this kind of corpora include the *Leeds Collection of Internet Corpora* and *CUCWeb* (cf. 1.1.2.1). Unlike the BNC (for whose creation permission was asked to the copyright holder of every single text) these corpora have not been copyright-cleared, mainly because "there is no easy way of determining whether the content of a particular [web] page is copyrighted, nor is it feasible to ask millions of potential copyright holders for usage permission" (Baroni et al. 2009: 227).

Therefore, in order to avoid breaching copyright laws, these corpora cannot be sold or distributed in any way. The solution adopted by many Universities to solve the problem of dissemination of these resources was creating online concordancers, i.e. web sites that only allowed users to see a small portion of the original web pages rather

²⁷ <http://goo.gl/l2n7z> (April 20th, 2012).

²⁸ <http://goo.gl/fud6K>

than distributing the corpora themselves to interested parties. This was not considered illegal because, as Fletcher (2004: 293) points out, at least "in the United States, a KWIC concordance of webpages appears to fall under the fair-use provisions of copyright law". The argument could also be made that, if displaying a small chunk of text taken from a web site were not fair use, then search engines would be breaking the law too (and on a much larger scale) because that is precisely what they do when they display a page of results (Kilgarriff & Grefenstette 2003). In this respect, search engines like Google or Bing go one step further in what might be considered unfair use of copyrighted material in that they also offer facilities to display a "cached copy" of web pages, that is a copy, stored on the search engine's own computers, of the full text of the page downloaded for building the index.

CHAPTER 2

SOFTWARE DEVELOPMENT

After presenting an overview of the most relevant corpus analysis tools currently available, in this chapter I will present the motivations for developing a new corpus manager that attempts to overcome some of the most crucial limitations of currently available software solutions. I will describe the system in some detail, concentrating on the features that set it apart from existing tools and focussing in particular on the advantages for end users and academic institutions who decide to adopt it for their corpus needs. I will then proceed to compare the new system to the desiderata expressed by Hoffman and Evert (2006) for an ideal corpus tools and I will conclude by outlining the current development status of the project and lay out plans for future work.

2.1 The requirements for a corpus manager

The goal of the project presented in this work is to build the foundations of a new corpus manager that overcomes the limitations encountered in currently available software. In this section I will present a set of requirements for the new system, some of which come from my own experience while others coincide with or were inspired by the "ideal corpus tool" proposed by Hoffman and Evert (2006), cf. also 2.5.

1. The system needs to implement at the very least all the core functionalities users have come to expect from a corpus query tool, like concordances, frequency lists, collocations and keywords.
2. The tool must require minimal training to use and no computer skills beyond those required to work normally with a software application, so it needs an interface capable of adapting to the user, i.e. very easy for neophytes but increasingly complex and powerful as users become more advanced. Since a certain degree of awareness of the problems specific to the methodology of the discipline is always desirable, this approach should not be taken to the extreme of not taking for granted a minimal competence in corpus linguistics.
3. The program must be available on all major platforms (Windows, Mac OS X and Linux).
4. The system must support multiple corpora with different kinds of annotation, at the token level (e.g. lemma, part of speech) and at the structural level (e.g. sentence, text, chunk, etc.).
5. Parallel corpora must be supported by the system.
6. Computer-savvy people must be given the possibility of building their own expansions to the system with relative ease and without the need to modify the core of the program.
7. It must be possible to make corpora available over the Internet (or a local network) while maintaining a fine-grained control on how much of these resources is visible: copyright is always a thorny issues when it comes to distributing corpora and the system must allow for an easy way to control who can access them and how (e.g. the quantity of context visible on a single concordance line, the possibility to see the full text from which the concordance was taken, etc.).

8. Users must be able to use their own corpora (i.e. upload them to the remote computer that hosts the service) and to decide whether they want to share them with other users or not.
9. Since the application is very likely to be exposed on the Internet, it needs mechanisms to prevent misuse or abuse, in particular, since corpus analysis tools are very resource intensive (i.e. even simple queries can rapidly exhaust the memory and processing power of even the most advanced computers) there must be a way of limiting a) the number of concurrent queries running on the server and b) the number of simultaneous queries submitted by each user.
10. Finally the application must be open-source in order to ensure the maximum degree of freedom in the development and distribution process.

The idea underlying these requirements is that of building a system that is as generic and modular as possible, with a server component capable of easily extracting minimally processed data from the corpus. The data can then be directly presented to the user in the form of concordance lines or frequency lists (which can also be exported), or they can be further processed by a fat client (cf. 2.6.3) via a set of plug-ins that allow one to extend the basic functionalities of the system.

2.2 Limitations of the current approaches

In chapter 1 several corpus managers were presented (cf. 1.1), therefore one could wonder if, given the great abundance of existing tools, there was a real need for a new one. I believe that there is indeed a need for it since none of them meets all the requirements laid out in 2.1, moreover I will argue that it is very difficult that any of the existing systems will ever be able to fulfil all of them, whereas the new approach I propose has a reasonable chance of doing it, at least in the medium and long

term and if enough interest can be generated around the software (which is necessary to increase the chance of attracting computer-savvy linguists who could contribute code and expertise to the project).

Moving on to a quick review aimed at explaining why existing software solutions do not meet the requirement, we could begin by pointing out that all off-line corpus tools (e.g. *Wordsmith* and *AntConc*) do not allow corpora to be made accessible via a network connection (requirement n. 7 of the list in 2.1) and are therefore not adequate solutions. As for the majority of the online systems presented in chapter 1, most of them were created ad hoc for a specific corpus, or set of corpora, and therefore cannot be easily adapted for use with different corpora. This is because when such ad hoc tools are built, many of the design decisions are based on the underlying data. For instance, when Andrew Hardie started the development of *CQPWeb* (1.1.2.1f. 1.1.2.1), he did it with the goal of creating a system that offered all the functionalities of *BNCWeb CQP-edition* (1.1.2.1n) but that was not limited to a single corpus. Despite the fact that he had access to the source code of *BNCWeb CQP-edition*, he decided to re-write the whole application from scratch using a different programming language instead of simply modifying the existing code because the process made it easier to understand where the code needed to be generalised (Hardie forthcoming). Another reason why existing online tools often cannot be modified is that the source code is not available for download or it depends on specific data or settings that are available on the computer running them and that cannot be easily replicated on a different machine.

Of course not all online corpus managers share these limitations, in fact some of them were designed precisely to be compatible with any corpora. The *Sketch Engine* (a commercial service based on Manatee, cf. 2.3.2.1) and *CQPWeb*, the official GUI of the Corpus Workbench, are two such corpus tools. Both are capable of supporting multiple very

large corpora (in the order of billions of tokens) with sophisticated multi-level annotation, but unfortunately neither of them meets all the requirements laid out in 2.1.

The *Sketch Engine* (Kilgariff et al. 2004) is an online corpus manager offering advanced functionalities and capable of supporting very large corpora. Unfortunately it is also a commercial service, a fact that has two important implications: firstly, users need to register and pay an annual subscription fee. Although this might not be a problem for academic institutions who could easily subscribe to the service, this could nonetheless be an impediment to a more widespread adoption of corpora outside universities, especially considering the low levels of awareness of non-academic people to the advantages that a data-driven approach can have on activities such as, for instance, professional translation (Bernardini & Castagnoli 2008). A second important consideration is the proprietary nature of the software: the Sketch Engine is a commercial service that cannot be customised, the company running the service has full control on the availability of corpora and on the software used to access them, therefore new features cannot be directly added when needed.

This all seemed to change in April 2011 with the release¹ of "NoSketch Engine" (Rychlý 2007), an open-source version of the Sketch Engine. Unfortunately many of the fundamental features of the Sketch Engine (such as keywords extraction and installation of custom corpora by users) had been deliberately removed from this new open-source version (cf. <http://goo.gl/sNgOF>), therefore it seemed unlikely that the development team would actively support any independent effort to re-implement free variants of those same features. Moreover, being a web-based system, NoSketch Engine shares many of *CQPWeb*'s limitations (see below). Finally, at this point development of *Corpse/Carcass* was in

¹ The availability of NoSketch Engine was announced on April 7, 2011 on the Corpora List, cf. <http://goo.gl/vRXqF> (March 10th 2012).

a fairly advanced stage so it would have been too late to incorporate it into my project.

CQPWeb (cf. 1.1.2.1) is another online corpus manager that supports multiple corpora and is available under an open-source license. It offers many of the functionalities expected in a corpus manager and meets the majority of the requirements outlined in 2.1. Still its approach has a few technical limitations, stemming mainly from the use of a web server as middleware (cf. 2.3.3), that cannot be easily overcome. For instance, in *CQPWeb* there is no way of preventing a user from submitting multiple queries at the same time: if the queries are computationally expensive it is possible to exhaust very quickly all the available computational power of the host computer. Note that this does not need to be done intentionally: when queries take too long (like in the case of computationally expensive operations) sometimes users just get tired of waiting for results and decide to hit the back button on the browser and submit a new query, or they close and re-open the browser window to try a different query while the first one is still running. This issue is not specific to *CQPWeb*, all web-based corpus tools have this limitation. Another problem of the web-based approach is that it makes it rather difficult to move part of the computational load away from the server: client-side processing could conceivably be done only using Javascript,² a programming language that does not have any mature communities devoted to building NLP tools (the only exception that I am aware of is the "natural" project³ which appears to be still in its infancy). Finally, it should be noted that at the moment, *CQPWeb* does not support parallel corpora.

2 N.B.: in order to avoid confusion, it should be noted that, despite the similarity in their names, Java and Javascript are two completely different and unrelated programming languages.

3 <http://goo.gl/QjdiT>

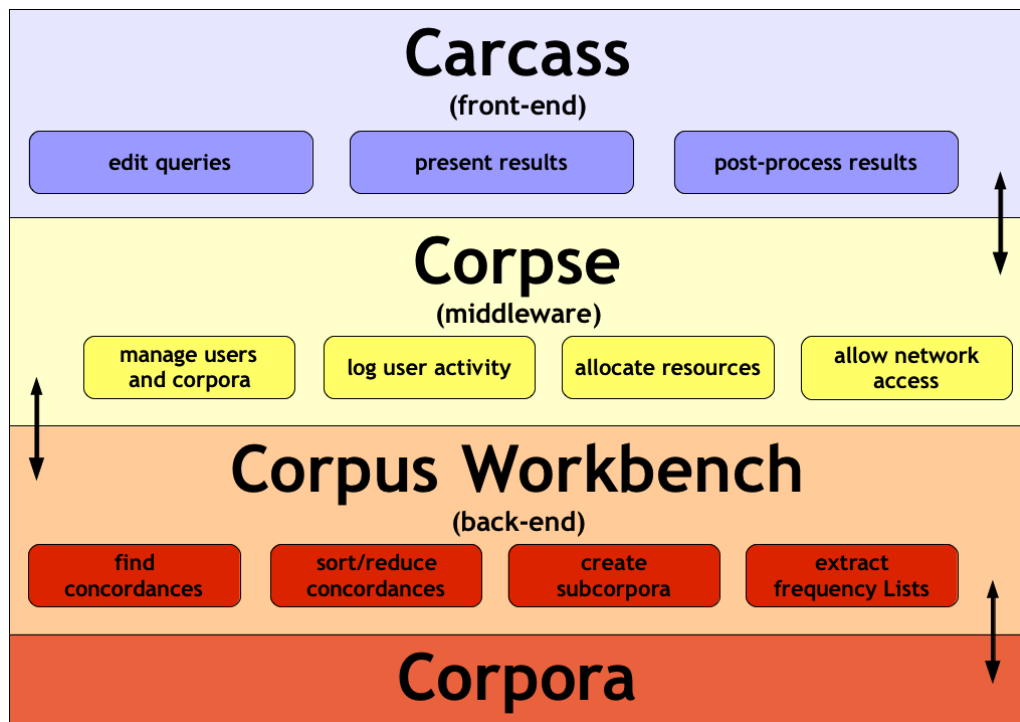


Figure 2.1: basic architecture of Sarcophagus

2.3 Project Sarcophagus

The system I set out to build uses the same *client-server* approach as the Sketch Engine and CQPWeb but with one very crucial difference: where the web-based systems use a standard browser as client and a generic web-server as middleware (cf. 2.3.3), all the software in the architecture I propose has been written from scratch specifically for managing corpora, keeping in mind the requirements of the users of this kind of resources.

2.3.1 Overview of the architecture

The new system I developed is formed by three main components. The base of the system is the IMS Open Corpus Workbench. On top of it rests the server-side application called *Corpse* (**Corpus Server**) which in turn supports the client application called *Carcass* (**Corpus Archive Search System**). The whole system is referred to as *Sarcophagus* (this

name is not an acronym). Figure 2.1 offers a schematic view of the system.

The IMS Corpus Workbench is an independent open-source effort carried out by Stevan Evert and Andrew Hardie (cf. 1.1.1.4) while *Corpse* and *Carcass* were written entirely by me.

2.3.2 The Corpus Workbench (the back-end)

At the root of the new *Sarcophagus* system lies the *IMS Open Corpus Workbench* (cf. 1.1.1.4), a set of command line tools widely used as a back-end for online corpus analysers (cf. 1.1.2.1).

The reasons for choosing CWB/CQP as a back-end were many; the first was the possibility to exploit the rich token-level (e.g. part of speech, lemma) and structural-level (e.g. text genre, author age, date of publication, etc.) annotation found in many corpora, such as the *BNC* and *Repubblica* for instance. Hoffman and Evert (2006: 180) claim that

the particular strengths of CQP are (i) the integration of an unlimited number of word-level annotations, document metadata and structural markup (in the form of XML start and end tags) in its queries; and (ii) the ability to perform very general searches (e.g. purely grammatical patterns such as noun phrases) on large corpora and efficiently handle the millions of hits they may return.

Another important factor was the availability of the software under the GNU GPL 3 open-source license, which meant that all the project's components (CWB, *Corpse* and the *Carcass*) could be freely distributed (cf. 2.1).

The query language was another important aspect in the choice of the system: this might seem surprising in the development of a user-friendly interface because the general consensus is that the CQP language is complex and difficult to learn, so much so that in recent years the developers of CWB created a simplified query language called CEQL for use in the BNCWeb (Hoffmann et al. 2008). Nonetheless, despite the fact that the CQP syntax is indeed rather complex and that

queries might look cryptic to the uninitiated, it is also true that motivated users can become proficient in it fairly quickly, albeit with a somewhat high error rate (see 3.1.6.1 for a description of how long it took to teach the basics of the language and 4.3 for a discussion on user behaviour, including error rates). Also, and this was perhaps the most important deciding factor, since CWB is so widespread (cf. 1.1.2.1) it makes sense to adopt its language since it has become a *de facto* standard in corpus analysis and many of the new system's potential users probably already know it.

Finally, since *Sarcophagus* is being developed at the Department of Interdisciplinary Studies in Translation, Languages and Cultures (SITLeC) of the University of Bologna, existing assets were taken into consideration when choosing the back-end. Most of the corpora available at the department are in CWB format, including *Repubblica* (Baroni et al. 2004), *EPIC* (Russo et al. 2006), *Comparapedia* (Bernardini et al. 2011) and the *Wacky corpora* (Baroni et al. 2009), to name but a few, therefore choosing a different back-end would have required the conversion of all corpora to a different format, a task that undoubtedly would have required a considerable effort and that could also have led to a whole series of unforeseeable problems since the corpora had been designed with a particular corpus manager in mind.

2.3.2.1 Alternatives to the Corpus Workbench

A few alternative back-ends were considered before opting for CWB. The first was Apache Lucene⁴ (McCandless et al. 2010), an open-source search engine that provides efficient full-text search over large quantities of text (in the order of billions of tokens, i.e. orders of magnitude more than CWB, which supports corpora no larger than 2 billion tokens). Unfortunately, the query possibilities of Lucene are not as extensive as those offered by CWB. One of the main reasons for

4 <http://lucene.apache.org/>

discarding Lucene was that while it is possible to use it for multi-level annotation at the token level, the query language does not allow users to put multiple constraints on the same token. For instance, it would have been possible to look for patterns like DETERMINER ADJECTIVE "cause" (which would return expressions such as "the underlying cause", "a lost cause", etc.), but not for patterns where "cause" was a verb followed by a noun, i.e. "cause"/VERB NOUN ("cause mayhem" or "cause problems"). The reason behind these limitations is that Lucene was designed to be a general-purpose search engine (e.g. like Google) and not to offer the kind of query functionalities an advanced linguist user expects. The possibility of adding multiple annotations on the same token was probably implemented just to permit an easy way of searching for synonyms or spelling variations. For example searching for "color purple" could return the same results as "colour purple" (which is the expected behaviour in a general-purpose search engine) because "color" and "colour" would have been indexed at the same corpus position.

Another alternative taken into consideration was the IXE search engine (Attardi 2005),⁵ which again supports multiple token-level annotation, has a powerful set of search features and an impressive performance. Unfortunately the software had to be discarded immediately because of licensing issues: the company who developed it was open to granting a free license to the University of Bologna, but not to release it under an open-source license (or as a freeware), which means that it would not have been possible to freely distribute the complete system (which was an important requirement in the development of the new architecture, cf. 2.1).

The final candidate for the back-end was the Manatee Corpus Manager.⁶ Manatee was very promising: it is open-source and presents

5 <http://www.searchtools.com/tools/ixe.html>

6 <http://www.textforge.cz/>

many of the advanced search facilities I was looking for, the query language is very similar to the CQP syntax and the system supports sophisticated queries like CWB. It also has a better performance than the Corpus Workbench.⁷ Unfortunately the system is poorly documented whereas CWB has a much more extensive documentation, a number of Perl libraries offering additional functionalities that can be directly used, studied and adapted or ported to a different programming language and a very responsive developer community. These are incredibly important factors when embarking in the task of writing software that depends on another piece of software: whereas lack of proper documentation can be a nuisance for regular users, it can be a powerful deterrent for developers who want to build a whole system on top of that program because they need to have a thorough understanding of how it works in order to exploit its functionalities.

Another important consideration is that bugs tend to emerge when pushing a program to its limits (something that happens very often when trying to anticipate users behaviour); when this happens, having access to a developer that can fix the code or give advice on how to work around the problem is invaluable. When I encountered CWB bugs during the development of Corpse/Carcass, the developers responded immediately and they were able to fix the problems in a matter of hours. When I hit a particular limitation of the software, they offered advice on how to solve the problem using a different approach. Another consideration was that, at the time of writing, Manatee can only run on Linux and Solaris, while CWB also runs on Mac OS X and Windows (at the moment the Corpse server component has only been tested on Ubuntu Linux and Mac OS X, but it could easily be adapted to run on Windows too). Finally, as I said above, all corpora currently available at the department where *Sarcophagus* is being developed are already in

⁷ <http://liste.sslmit.unibo.it/pipermail/cwb/2007-February/000058.html>

CWB format and converting them would entail at the very least a considerable effort.

All in all, it was felt that none of these candidates offered any substantial advantage over the Corpus Workbench.

2.3.2.2 *The role of the back-end*

But what role does the Corpus Workbench play in the system? As can be seen in Figure 2.1, the back-end is the part of the system that directly interacts with the corpora, here the actual data are examined: the various CWB tools (namely CQP and `cwb-scan-corpus`) receive instructions from Corpse to perform operations such as finding concordances or compiling frequency lists. Once the data have been retrieved from the corpus, they are passed on to Corpse for further processing.

2.3.3 Corpse (the middleware)

Corpse is the server component of Project Sarcophagus, the technical definition for this type of software is middleware, an "extra layer" between the client (i.e. the program users install on their computers and with which they interact directly) and the back-end (in our case CWB, that is the program that actually queries the corpora). Corpse is a key piece of the system because it facilitates interaction between the client (or front-end) and the back-end and makes it possible for the various components to communicate with each other over the network. This type of software, in the context of a *client-server* architecture (Sarcophagus is one such architecture), is often also called a *server*.⁸

⁸ Since the term "server" can also be used to refer to a computer that simply runs a server application, in order to avoid confusion in this work I will always use "server" to refer to an application, whilst when referring to the actual computer that runs a server application I will use the term "computer" or "host".

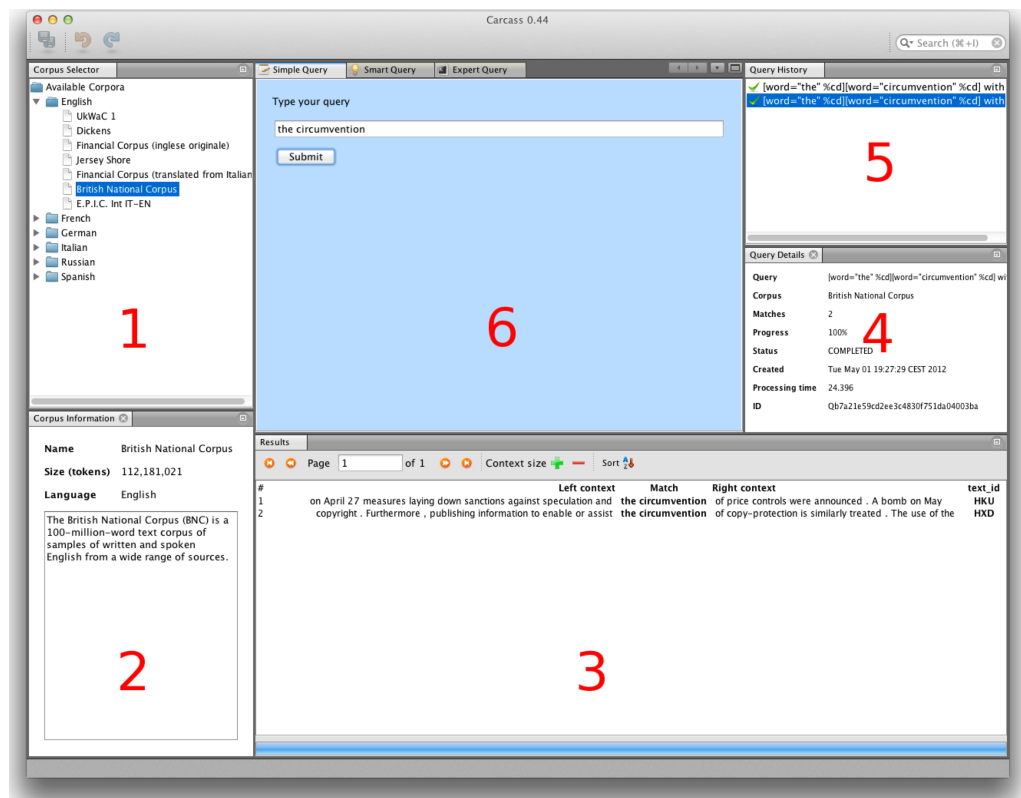


Figure 2.2: main screen of Carcass

2.3.4 Carcass (the front-end)

Carcass is the front-end, or client, the program that allows users to interact with the system. Through Carcass it is possible to obtain information about available corpora, submit queries, display and manipulate concordance lines, browse the query history and export results.

Figure 2.2 shows the main screen of Carcass in which several distinct panels, marked by the numbers 1 through 6, can be identified. Panel 1, the "Corpus selector", shows the corpora, grouped by language, that are available on the server. Panel 2, called "Corpus information", contains details about the currently selected corpus (name, size, language and a short description). In panel 3 ("Results") the results of the currently selected query are displayed, while panel 4 ("Query details") presents additional information about the query, like number of matches or processing time, it also contains the query in CQP format.

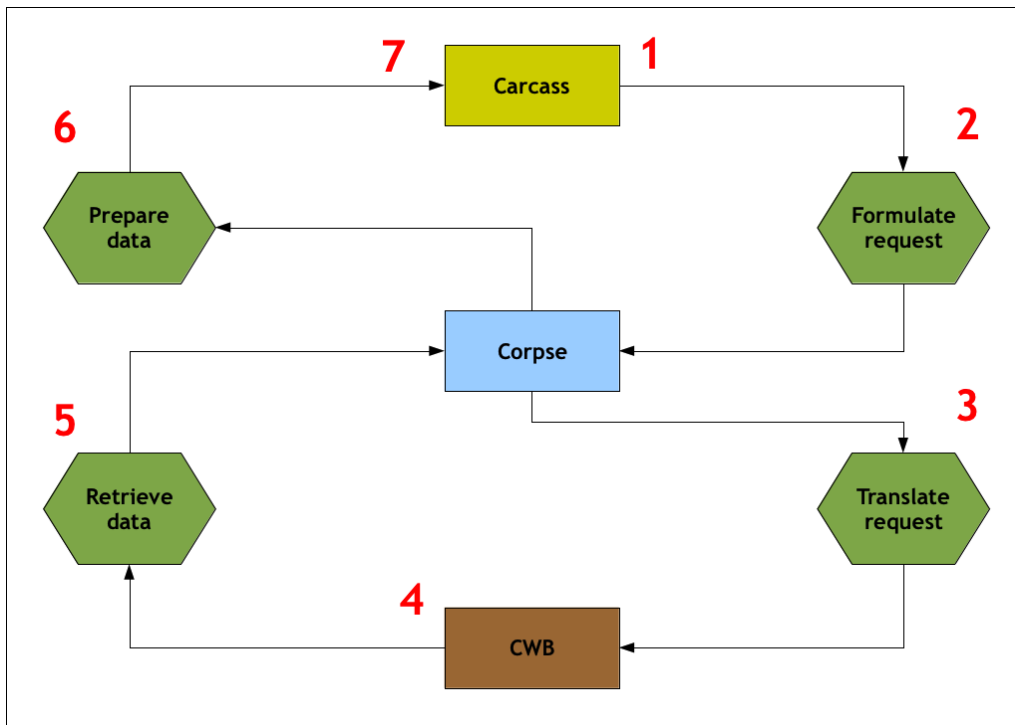


Figure 2.3: lifecycle of a typical user interaction (simplified version)

Since queries that are submitted using the "simple" or "smart" editors (cf. 2.4.1.1 and 2.4.1.3) are always translated into the CQP query language, they are displayed here in that format.

All queries submitted during the current session are shown in panel 5, "Query history", from where previous queries can easily be recalled and displayed. It is also possible to use the operating system's *copy* facility (e.g. pressing CTRL-C on the keyboard) to copy the text of the query to the clipboard, which can be very useful to modify queries formulated with the "smart editor" (cf. 2.4.1.3). Finally panel 6 hosts the three main query editors, tools that allow users to formulate queries employing different strategies (cf. 2.4.1).

2.3.5 How the system works

Figure 2.3 shows a simplified diagram (see 2.3.6 for a more detailed explanation of the process) of what happens in a typical user interaction

with the system (e.g. submit a query, sort results, change context size, etc.).

Consider this typical scenario: a user has performed a query, is browsing the obtained concordances and now wants to sort the results. The user clicks on a button in Carcass (step 1) which formulates a specific, high-level instruction (step 2, e.g. "sort results of query X by right context, ignoring letter case and diacritics") and sends it to Corpse which translates the request to the formal CQP language (step 3, i.e. "sort by word %cd on matchend[1]") and passes it on to CWB. The Corpus Workbench performs the operation (step 4), retrieves the results (step 5) and returns them to Corpse, which in turn prepares them (step 6) and forwards them to Carcass which can then present them to the user (step 7).

2.3.6 A more in-depth look

The diagram presented in Figure 2.3 (page 36) is obviously a simplification of how the various parts of the architecture interact with one another. Without going too much into the details of the inner workings of the system, I will now focus on one particular problem I had to face when developing the architecture, synchronisation, explaining why this is such an important issue.

One of the aspects that Figure 2.3 fails to capture is the timing of the operations: CQP queries can take quite a lot of time, especially when the first token in the query has a high frequency in the corpus being searched.⁹ Consider for instance the case of the following query, designed to extract noun-noun compounds in English¹⁰:

(a) [pos="NN.*"] [pos="NN.*"]

⁹ On the issue of performance, see also comments by CQP developers Stefan Evert and Andrew Hardie available at <http://goo.gl/m5k6I>.

¹⁰ All queries in this work refer to corpora automatically tagged with the TreeTagger (Schmid 1994); English corpora use the Penn Treebank Tagset (Marcus et al. 1993) while Italian corpora use the Repubblica Tagset (Baroni et al. 2004).

Since there is bound to be a large number of nouns in the corpus, the first item in the query will have many hits and the query will be slow.¹¹ On the British National Corpus (112 million tokens)¹² this query took 23.7 seconds and returned 2.355.571 matches. Now if we consider that the time required to complete a query increases proportionally with corpus size, it should come as no surprise that the same query, when run on UkWac (2.1 billion tokens) took 440.4 seconds to complete (in this case 57.831.165 matches were found). Looking at these figures it becomes evident that in many cases results cannot be returned instantly as Figure 2.3 might seem to suggest (or as users accustomed to the instantaneous response time of search engines might expect).

In the light of these facts, we could offer a slightly more accurate description of what actually happens in the interaction illustrated in Figure 2.3: after step 4, CWB does not send the results back to the client, it simply starts processing in the background, while Corpse sends an acknowledgement back to Carcass saying that processing is underway (and also giving an estimate of the completion time of the operation, which allows Carcass to show a progress bar). Corpse keeps checking on CWB to see if the operation has completed and updates the status of the request. Carcass in turn checks regularly with Corpse to get a progress report. When the operation has completed on CWB, Corpse caches the results and, the next time Carcass checks in, tells it they are ready. Only when the results are with Corpse can Carcass call in again and retrieve them to finally present them to the user.

This is a much more sophisticated approach than the one taken by typical web-based systems (such as CQPWeb) which basically use a web server as middleware. The approach taken by Sarcophagus offers many advantages (cf. 2.4.1), two of them are easily illustrated: first of all, in all

¹¹ All tests were conducted on a Dual Xeon E5405 @ 2.00GHz with 8 GB of RAM, a fairly modern and powerful computer.

¹² This is the size of the corpus as reported by CQP, which also counts punctuation and parentheses as tokens.

the online corpus managers I examined (including the Sketch Engine, CQPWeb and the BYU corpora), users have to wait for processing to finish before they can do anything else. In most cases, this is not a problem because processing is very fast, but as we have seen above, sometimes queries take a long time to finish. In Carcass it is possible to browse the query history or start writing a new query while processing is underway. Additional queries can also be submitted while waiting for previous ones to complete, if the user has permission to use more than one process, then additional queries will start immediately, otherwise they will be queued and execution will start as soon as the previous ones have completed.

A second advantage of the client-server approach adopted in Sarcophagus is that users can immediately see the result of their actions: as soon as the query is submitted a progress bar appears to inform them of how long the operation will take. From the point of view of usability, when operations require more than a couple of seconds, it is important to provide users with some sort of feedback to reassure them that the program is working and possibly give an estimate of the time it will take to complete the operation (Nielsen 1994). Even though it is possible to implement this kind of feedback also in web-based system, none of those I examined (cf. 1.1.2) do it and users have to wait for processing to finish while staring at a seemingly unresponsive web page.

2.4 Advantages of the Sarcophagus system

It is my opinion that the more "traditional"¹³ client-server model adopted by the Sarcophagus project presents several advantages, both for end-users and for corpus builders that want to disseminate the results of their work. These advantages stem from the fact that, once the basic plumbing of the system is in place and the various components

¹³ Traditional in the sense that both the client and the server are *ad hoc* applications, as opposed to other approaches that leverage on existing general-purpose software such as a browsers and web-servers.

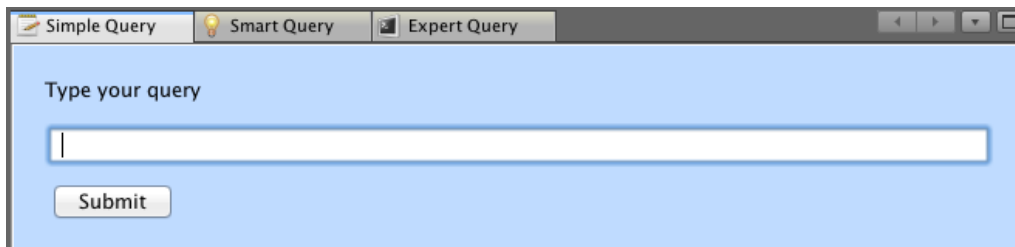


Figure 2.4: Simple Query Editor

can communicate with each other, it is relatively easy to add new features to the system, mainly thanks to the adoption of a modular platform like Netbeans¹⁴ (Petri 2010) for the client.

In the following sections I will present the most notable features of Sarcophagus from the point of view of the users of the system and from that of the corpus builder/system administrator.

2.4.1 Sarcophagus for users

Carcass is the front-end, the tool used to interact with the system, therefore the usability of the whole architecture depends mostly on this component. Having an ad-hoc client that runs on the user's computer instead of a web-page generated by a remote server makes it possible to have a very rich and reactive interface: since the program runs directly on the user's computer, the interface is drawn and updated locally so there is no need to wait for the web server to generate the page and for the browser to download it every time the user moves to a different part of the application. Rich and responsive web interfaces are arguably possible using AJAX based frameworks like Ext JS¹⁵ (Frederick et al. 2010) or the Google Web Toolkit¹⁶ (Kereki 2010) but none of the available corpus managers I examined uses this kind of technology.

Figure 2.2 (page 35) shows the main window of Carcass, in the panel marked by the number 6 are hosted three different tools, called

¹⁴ <http://netbeans.org/>

¹⁵ <http://goo.gl/ZqwWP>

¹⁶ <http://goo.gl/157F1>

"Simple Query", "Expert Query" and "Smart Query" which allow users to compose queries employing three different strategies.

2.4.1.1 Simple Query Editor

The *Simple Query* editor is just a plain text box (see Figure 2.4) where users can type one or more words, no special syntax is required in this context, the program takes care of "translating" the query into the CQP language. By default, all simple queries ignore case and diacritics and are prevented from crossing sentence boundaries (i.e. all words in the phrase must be in the same sentence). For instance, typing "cat" in the *Simple Query* editor results in the following CQP query:

```
(a) [word="cat" %cd] within s;
```

A simplified version of the Kleene Star operator (i.e. the "asterisk") is also supported: normally the CQP syntax requires that the pattern "zero or more characters" be expressed using a dot followed by a star. For instance looking for words beginning with "under" would require this CQP query:

```
(b) "under.*";
```

In the Simple Query Editor users can instead simply type:

```
(c) under*
```

Although this feature appears to improve the usability of the *Simple Query*, analysis of the query logs (cf. 4.3.2.1) suggests that once users learn this syntax, they tend to use it also in the other query editors where the feature is not (and cannot be) supported. This leads to subtle errors that can be misleading, in fact a CQP query such as

```
(d) "under*";
```

is perfectly valid, but it means: look for all occurrences of "unde" followed by zero or more "r". In future versions of Carcass the simplified use of the Kleene star will probably be removed in favour of a "live" spellcheck (i.e. a warning will appear as the user is typing) that will

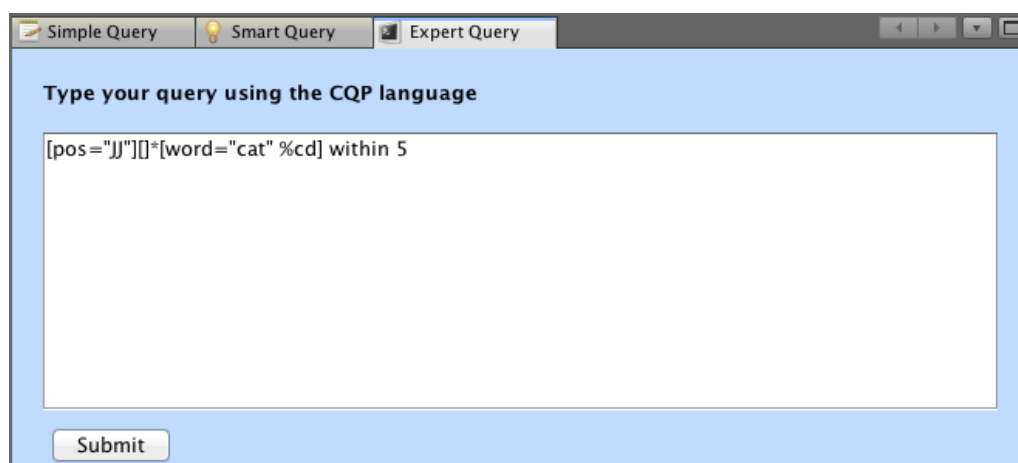


Figure 2.5: Expert Query Editor

advise users to add a dot right before an asterisk. This will maintain intact the usability of the *Simple Query* while educating users to the correct syntax.

2.4.1.2 Expert Query Editor

The *Expert Query* editor (Figure 2.5) is a slightly larger text box that offers users the possibility of using the CQP syntax to its full extent. At the moment there are no particular features in this tool beyond the basic text editor, but there are plans to implement syntax colouring, error highlighting and code completion for the CQP language. All this should be relatively simple to add, given that the Netbeans platform (on which Carcass is built) already offers support for a variety of formal languages and can easily be extended to support new ones.¹⁷

2.4.1.3 Smart Query Editor

These first two editors we examined are pretty standard, but the "Smart Query Editor" is different from the usual choices available in corpus managers¹⁸ in that it allows the use of a combination of drop-down menus and checkboxes that simplify the task of composing complex

¹⁷ <http://goo.gl/TOH6V>

¹⁸ The Smart Query Editor was inspired by CucWeb's "Expert search", cf. <http://goo.gl/rRiAD>

The screenshot shows the 'Smart Query' tab of the Smart Query Editor. It contains three identical query entry sections, labeled 'Position 1', 'Position 2', and 'Position 3'. Each section has three input fields: 'Word is', 'Lemma is', and 'Part of speech is'. Below these fields are two checkboxes: 'position is optional' and 'ignore case and diacritics'. In Position 1, all fields are empty except for 'Adjective' in the dropdown. In Position 2, 'position is optional' is checked. In Position 3, 'cat' is entered in the 'Lemma is' field. At the bottom of the form are 'Clear form' and 'Submit' buttons.

Figure 2.6: Smart Query Editor

queries. The query illustrated in Figure 2.6 would be "translated" by Carcass into the following CQP query:

```
(e) [pos="JJ.*"] [pos="JJ.*"] ? [lemma="cat"]
```

and would return matches like "large tabby cat" and "tiny cats".

The drop-down menus in the part-of-speech box contains both "human readable" items (i.e. "Adjective", "Noun", etc.) and plain tags ("NP", "JJR", etc.). The more "human readable" items often group together more than one tag (cf. 2.4.2), for instance when "Adjective" is selected, the choice is translated into "JJ.*" a regular expression that matches three different tags: JJ (regular adjective), JJR (comparative adjective) and JJS (superlative adjective). It is of course also possible to select individual tags from the list (i.e. just "JJR" or just "NNS"). The "ignore case and diacritics" checkbox is pretty self-explanatory, while the "position is optional" checkbox is used to indicate that the token described in that position is optional, it can either be there or not: query

(e) returns "cat" preceded by one or two adjectives because the second adjective (the one in position 2) is optional.

This editor allows sequences of a maximum of three tokens, the choice was dictated mainly by reasons of space in the interface, if user feedback should indicate that more positions are required they will be added to the form.

2.4.1.4 Other features

The *Query History* is the panel in which all submitted queries are stored, it is marked by the number 5 on Figure 2.2 (page 35). From here it is always possible to recall the results of previous queries without having to re-submit them. Users can also copy queries from here and paste them in the "Expert Query Editor", which is useful for refining queries submitted through the *Simple* or *Smart* editors (cf. Appendix H).

The *Corpus Selector* (panel 1 of Figure 2.2) allows users to select the active corpus and makes it very quick to switch from one corpus to the other. Currently corpora are only grouped by language but in the future it will be possible to group them according to other criteria (cf. 2.4.2).

Right below the *Corpus Selector* we find the *Corpus Information* box which provides basic information on the currently selected corpus, like language, number of tokens and a short description. This box is currently hidden by default (it can be activated via the *Window* → *Infobox* menu) because the information it provides is not essential and during early tests of the application it seemed to confuse users who were presented with too much information. In future version, the details about the currently active corpus will probably be presented in a different way.

Below the *Query History*, the *Query Details* panel can be found, which presents additional information on the currently active query

(number of matches, progress status, processing time, etc.). This panel is hidden by default too (it can be displayed via the *Window* → *Query Details* menu), for the same reasons the *Corpus Information* box was hidden: too much information appeared to be detrimental to the user experience, this panel will probably be kept in future versions but most of the details currently presented here will be removed (e.g. QueryId, Processing Time).

This concludes the overview of the main features available to users of Carcass, a hands-on tutorial which can be used to get acquainted with the program can be found in Appendix H.

2.4.2 Sarcophagus for corpus builders

The new system allows users to access corpora using Carcass, a friendly and reactive interface, but most of the work of project Sarcophagus was devoted to the development of Corpse, the middleware (cf. 2.3.3), the glue that holds together the various components of the system. In this section I will present some of the features that will make it easier for corpus builders, especially Universities, to make their corpora more easily available to students or scholars.

2.4.2.1 Fine-grained permissions

One of the main issues anyone wishing to build a corpus has to face is that of copyright (cf. 1.2), consequently the ability to control who has access to which resources is paramount. This is where a fine-grained permissions system comes in: Corpse allows system administrators to define an arbitrary number of groups, each corpus is assigned to one or more groups and each user connecting to the server is also assigned to one or more of these same groups. Users will only be able to see the corpora that belong to their assigned groups.

For instance, consider the diagram in Figure 2.7: User 1 belongs to Groups A and B and thus will be allowed to see Corpus 1, 2, 3 and 4,

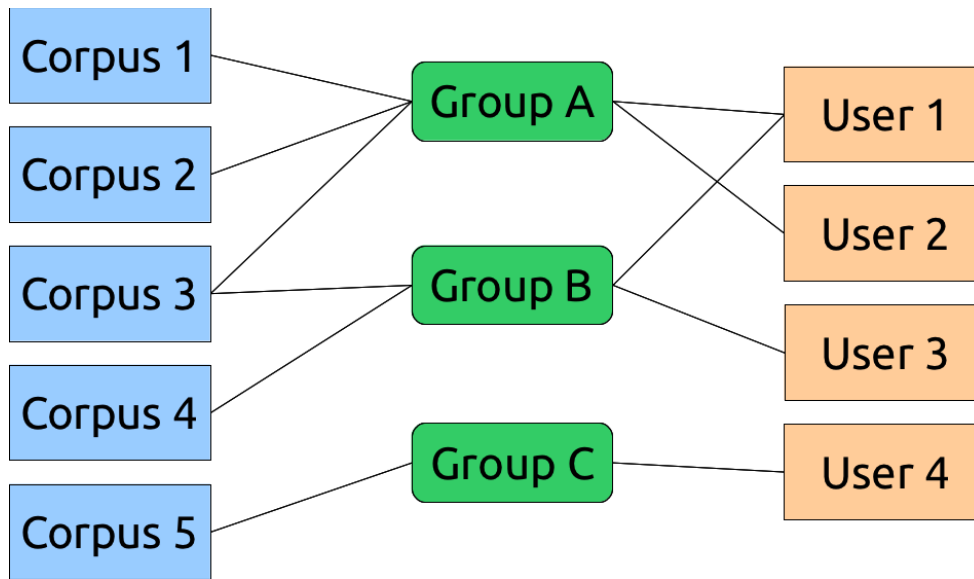


Figure 2.7: permissions

whereas since User 3 belongs only to Group B, he or she will only be able to see Corpora 3 and 4. When a new corpus is added to the system, the administrator decides to which group(s) it will belong via a simple configuration file, there is no limit to the number of groups to which each corpus can belong as there are no restrictions to the name of the groups.

When users connect to the server via Carcass they have two options:

1. connect simply without providing any credentials (i.e. username and password), in this case the system performs an IP-based authentication;
2. provide username and password, in this case the credentials-based authentication is enforced.

The IP-based authentication assigns users to one or more groups based on the subnet¹⁹ they connect from: for instance, users connecting from computers located within to the University of Bologna (i.e. belonging to the 137.204.0.0/16 subnet) are assigned to groups

¹⁹ The subnet is determined by the IP address of the computer they are using.

"restricted" and "unrestricted" and thus have access to corpora belonging to those groups, whereas clients connecting from outside this subnet (i.e. 0.0.0.0/0) are assigned only to group "unrestricted". It is also possible to assign different permissions to smaller subnets, for instance subnet 137.204.200.0/24 (which is a subset of the University of Bologna subnet) could be given a different set of permissions from those of the higher level subnet. This makes it possible to give special access to users connecting from a particular lab for instance, which is particularly important because permissions also determine the number of simultaneous queries that can be executed by the client: in this way a research lab could be given more processing power on the server than a room where students are just familiarising themselves with corpora. As a safety measure, IP-based authentication also allows completely blocking access from subnets, something that could prove useful to avoid abuses from rogue computers.

Credentials-based authentication works in much the same way as IP-based authentication with the only difference that permissions are assigned on a per-user basis. This allows members of an organisation to connect to restricted resources even when they are connecting from home for instance, or to grant special permissions to users who accept certain terms and conditions, as in the case of corpora that can only be accessed for academic and non-commercial use.

2.4.2.2 Configuration

Corpse tries to simplify the work of system administrators by doing most of the configuration automatically: when the server starts it detects all corpora available on the computer and, if instructed to do so, generates lists of positional and structural attributes that can be used by the client to generate lists and menus (for instance, all plain tags lists in the *Smart Query Editor* are generated automatically); this reduces the work of the system administrator, who does not have to manually

compile them, and makes sure that the values in the pick lists are the ones the corpus actually uses.

Each corpus can have a custom configuration, in the form of a plain-text file, which defines a few key parameters like the groups it belongs to and the name of the positional and structural attributes that should be used to compile the pick lists. In case a corpus has no custom configuration file, it is automatically assigned to the "orphans" group which is hidden to all users by default: this is a safety measure put in place to avoid inadvertently exposing restricted content on the Internet.

The features presented above make it much easier to deploy new corpora efficiently and safely, this is advantageous not only to system administrators but also, indirectly, to users who can gain access to resources that would otherwise be precluded to them because of technical and legal complications.

2.5 Sarcophagus vs. Utopia

Hoffman and Evert (2006) propose a list of characteristics that a corpus tool should possess in an "ideal world" to satisfy the needs of linguists.²⁰ In this section I will try to determine how the Corpse/Carcass system fares when compared against this list. Since some of the requirements they propose depend on the back-end (over which I have no control, cf. 2.3.2.2), I will only consider those aspect that depend on the part of the system I developed.

Hoffman and Evert wish for their ideal corpus tool to possess "highly intuitive and user-friendly query specification" (ibid.: 181) that even novice users can easily master. While Carcass is far from being usable by novice users without any training, in the light of the data presented in 4.3.1 it seems fair to say that this goal has been at least partially accomplished by offering users the choice between three

²⁰ It is worth noting that Stefan Evert is the main developer of the IMS Open Corpus Workbench.

different query editors, dubbed *simple*, *smart* and *expert* (cf. 2.4.1). These three levels present an increasing degree of sophistication and help novice users transition smoothly from plain queries to more advanced ones. There are also plans to make the expert query much more user-friendly even for advanced users (cf. 2.6.1).

Hoffman and Evert also advocate for a tool that has no restrictions on the specific corpus and that allows new corpora to be added with minimal effort. This requirement has also been partially met by *Sarcophagus*: on startup, Corpse scans the host on which it is running for any CWB corpus and makes them instantly available for any client connecting to it (provided they have permission to access them of course). Unfortunately, at the time of writing, there is no easy way to add a new corpus to the system (cf. 2.6.3), the only way is to install it manually using the standard CWB command line tools. Once the new corpus is available on CWB, it will also become available via Corpse on the next server restart.

An attempt has been made to implement a "flexible and intuitive display of query results (with extended context); the complete set of corpus annotations relevant for each match can be conveniently accessed" (ibid.); unfortunately the way results are displayed in Carcass was not particularly appreciated by users (cf. 4.2.1) and at the moment corpus annotations cannot be displayed at all.

Hoffmann and Evert advocate for a platform independent solution that requires no advanced computer skills (ibid.: 182). This is certainly true for end users of the *Sarcophagus* system who only need to download and run the Carcass client. The same is not true for system administrators who want to install Corpse on a host in order to make one or more corpora available to the public: at the moment there is no installation script for Corpse and setup on a new computer has to be done manually.

The final requirement is for a "stable and robust implementation that does not crash even on faulty input and supports a large number of concurrent queries without exhausting the server computer's resources (such as memory, disk space and CPU time)" (ibid.: 182). This has been partially accomplished by Sarcophagus: the experiment described in 3.1.6 required the simultaneous connection of dozens of users at the same time, and although a few clients did crash, the server did not and most users were able to complete the task without interruptions and with no loss of data.

Hoffmann and Evert also mentioned requirements regarding post-query features, none of them have been met because no post-query features have been implemented in Sarcophagus yet. Although it is worth mentioning that the adoption of the Netbeans platform makes the creation of custom plug-ins very easy, which should make it possible to fulfil the requirement for a way for computer-savvy linguists to create their own modules for post-query processing (ibid.: 181).

2.6 Future plans

The new system presented in this work is far from complete, in fact the first version, which will be the object of the evaluation proposed in chapters 3 and 4, has not been released publicly yet and has shortcomings in many areas. A number of improvements and additions are planned for Sarcophagus and in this section I will outline the plans for the system in the short, medium and long term.

2.6.1 Short-term developments

The improvements planned in the short-term are aimed at solving the problems highlighted by the evaluation and adding a few fundamental features that could not be included in the current version for lack of time. All this will be done towards the goal of creating a beta version which can be released publicly.

One of the greatest weaknesses highlighted by the evaluation I carried out (cf. 4.2) is the way concordance lines are displayed: the current implementation of this functionality is arguably rather confusing and few users expressed appreciation for it. A rewrite of the results visualisation component will be one of the first items on the agenda for future improvements of Carcass.

More in general the interface needs to be streamlined: in 2.4.1.4 I emphasised how the *Query Details* and *Corpus Information* components need to be changed, much of the information presented in this panels is either unnecessary (e.g. query id, query status) or has a very low priority (e.g. processing time, creation date) and can thus be removed or hidden by default, only to be revealed when users explicitly look for it (Lidwell et al. 2003).

Another important feature that has been completely left out in this first release is the possibility to query aligned parallel corpora, this is a very important aspect of the system (cf. 2.1) that will be implemented in the near future: the Corpse server component already has full support for aligned parallel corpora, the only thing missing is a clean way of integrating the functionality in the front-end. This was left out from the current version for lack of time and because it was not necessary for the first stage of the evaluation I propose in this work.

Corpse also already implements a rich set of functionalities to extract frequency lists from corpora (in fact, this is how positional attributes pick lists are generated in the automatic configuration step, cf. 2.4.2.2). Again the only missing piece is the integration with the user interface, so this feature will see the light very soon. The support for structural annotation in Corpse is also complete, so it will be very easy to add filters in the front-end that allow to restrict searches according to metadata for corpora that include them (like *Repubblica* and *EPIC* for instance).

Finally, a feature that will be slightly more complex to implement but one that will make a big difference in terms of usability (so it should definitely be included in the first public release of the software) is a full parser for the CQP language, including a syntax highlighter with autocomplete and spell-check functionalities.

2.6.2 Medium-term plans

In the medium term, more work will be done on post-processing results, that is the possibility of adding more types of analysis to the mere browsing of concordance lines. In 2.1 I stated that the main goal of this new system is that of providing a platform for the extraction of minimally processed data from corpora (i.e. frequency lists and concordance lines) that could be used to perform other types of analysis, like extracting keywords, computing co-occurrence to find collocations, extracting terms, etc. All these operations could be done by the client, so they would not have an impact on the performance of the server in case multiple users try to perform these operations at the same time (this happens very often, for instance, during corpus linguistics classes, when dozens of students try to extract collocates at the same time).

Another medium-term goal is that of allowing users to submit *subqueries*. In CQP parlance, subqueries are searches performed within the results of a previous query, for instance is someone were to be interested in studying the word "pole" they might submit a query like this:

```
(f) [lemma="magnet.*"]
```

and then search, within the results of that first query, for:

```
(g) [lemma="pole"]
```

in this way it would be more likely that they would obtain results like "magnetic pole" or "North Pole" instead of "pole vaulting" or "bamboo poles".

2.6.3 Long-term prospects

Further in the future, if enough interest in the project will have been generated, Sarcophagus could permit users to upload their own corpora as raw text files that the system would automatically tag using the TreeTagger. Integration of multimedia elements could also be considered, since there are examples of corpora linked to multimedia elements (e.g. EPIC cf. 1.1.2.1) that are currently very hard to use and not publicly available.

Smith et al. (2008) advocate for the possibility to add custom data to query results and they propose that this be done by exporting data from the corpus tool into existing, general-purpose programs like the *Microsoft Excel* spreadsheet or the *Filemaker Pro* database. This is certainly a viable solution (*BNCWeb* in fact implements this feature using this very approach, cf. 1.1.2.1), but one that requires using different software applications that cannot easily communicate: the data would have to be moved between the two applications in a shared format, a process that could be tedious and difficult to accomplish for people with limited computer skills. I would argue that it would be preferable to integrate this kind of functionalities directly into Carcass: this would allow easy data manipulation since everything would be accomplished in the same application and without the need to rely on external programs. An even more interesting idea would be that of supporting crowd-sourced annotations directly to the corpus that could be made available to other users of the platform. This is a very stimulating prospect that should be discussed directly with the CWB development team.

A tool dedicated to system administrators would be also desirable, currently all configuration is handled manually via text files and direct manipulation of data contained in a MySQL database. The configuration of the system is not particularly complex but if the system were to be adopted by other institutions, such a tool would become a priority.

Finally it is worth mentioning that it is also conceivable that the whole Sarcophagus architecture (both the client and the server components) could one day be distributed as a stand-alone package that users could download and install on their own computers: users would basically install a server on their local computer that would only be accessible to them. Then, using Carcass, they could either connect to their local server (on which they could install their own private corpora) or to a remote server.

2.7 Availability

Carcass is available for download on the official website of the project;²¹ at the time of writing, version 0.44 is the most recent release of the program. The Corpse server component has not been published on the website because installation instructions have not been prepared yet, but as soon as an installation guide is ready, it will be possible to obtain Corpse too. The website also contains a "Help" section, which includes two hands-on tutorials (see Appendix H), a troubleshooting guide and some Frequently Asked Questions.

Since all software in the project is released under the GNU General Public License version 3,²² the source code of Corpse and Carcass can also be downloaded from the website.

²¹ <http://sarcophagus.sslmit.unibo.it/>

²² <http://goo.gl/rzSCv>

CHAPTER 3

EVALUATION METHODOLOGY

After having described the system in detail, in this chapter I will describe an experiment designed to test the viability of Sarcophagus as a corpus query tool by comparing it to CQPWeb (1.1.2.1).

CQPWeb is a web-based corpus analysis system (Hardie forthcoming), designed as a clone of BNCWeb CQP edition (Hoffmann & Evert 2006). Apart from the technical differences (CQPWeb is written in the PHP programming language while BNCWeb uses Perl), the main advantage of CQPWeb is that, unlike BNCWeb which can only be used to query the British National Corpus, it is compatible with any corpus. CQPWeb has been designated as the official interface of the IMS Open Corpus Workbench, it is open source and can be downloaded from the OpenCWB web page.

CQPWeb is obviously a much more mature software than Carcass, has a wider set of features (e.g. collocations, various post-processing features, etc.) and has been available since 2009 (the first version available for download on Sourceforge is 2.02, released on June 13th, 2009).¹

The idea behind this proposed evaluation was not to determine whether Carcass was a better or worse interface than CQPWeb, rather its

¹ <http://goo.gl/jrLGC>

purpose was to find out if Carcass (which is still in its infancy) was on the right track and the limited set of features currently implemented could measure up with the corresponding features of a more mature system. Even more importantly, a by-product of this evaluation process is the possibility to directly observe how users actually interact with corpus tools in a controlled environment. Results of the experiment will be presented and discussed in chapter 4.

3.1 Experiment description

The experiment was intended to observe how a group of users actually interact with corpora. Since I had access to a large number of students in translation, the logical thing to do was to devise a task that would emphasise the role of corpora in translation-related tasks.

3.1.1 The original plan: a translation task

The original plan was to have a group of students translate a technical text without the aid of dictionaries, terminology databases or online resources such as Word Reference or IATE, the only thing they could use would be a small set of specialised corpora. This plan was quickly discarded for a number of reasons.

Firstly, since the idea was to have students look up terms and expressions they did not fully understand in English or did not know how to translate into Italian, it would have been necessary to find a real text (or a portion of text) that contained a sufficient number of terms or expressions typical of a particular domain and that posed a real challenge to them. In order to have as realistic a scenario as possible, the text would have had to be a real one and not one made artificially more difficult, for instance by combining sentences found in different texts and/or adding specialised terminology or expressions. Finally, the text would have had to be relatively short because it would have had to be completed in the space of a single class (90 minutes). Moreover,

since the students I had access to are accustomed to being assigned translation tasks and are usually required to produce high-quality texts in the target language, it was very likely they would spend a considerable amount of time refining the target text. This normally desirable aspect of the translation process would subtract time from what was the focus of the experiment (i.e. having students use corpora).

More in general, this approach would have left very little control over all the variables involved in the experiment: I wanted to be able to control which of the two systems (Carcass or CQPWeb) was to be used at any given time during the experiment and I wanted to make sure that both systems would be used for the same amount of time and on the same number of terms/expressions. Leaving students free to pace their work as is the norm during a translation task would have made this very difficult. Likewise, imposing a time limit (e.g. instructing them to use one system during the first 30 minutes and the other during the following 30 minutes) would not have guaranteed an equal treatment of the two systems because the density of terminology in the text was bound to be uneven.

3.1.2 The actual plan: a documentation task

For all the reasons outlined above, it was decided that a traditional translation task would not be adequate and that a documentation task would be more appropriate: participants would be asked to use a set of specialised corpora to find Italian terms and expressions equivalent to the English terms and expressions present in a real text.

Students were given a text (in English) in which a set of expressions were highlighted and using only one system (Carcass or CQPWeb) their task was to:

- find the equivalent expressions in Italian;

- find an example of its use which was similar to the English example they had been given;
- explain the corpus-based strategies they used to find the equivalent;
- explain why they chose that particular example.

The text from which the expressions were taken was the "Annual report and accounts 2010" of EasyJet, one of the UK's largest airlines (an excerpt of the text can be found in Appendix B). This text had been independently chosen by the teacher as the main subject of the translation course. The experiment took place at the very beginning of the semester, when the course had just started (see 3.1.4).

3.1.3 Resources

In a similar study, Hafner & Candlin (2007) left students free to use whatever resources they wanted and tracked the activities of participants only when they used the tools that were the object of the experiment. This had been the original idea for the present experiment too, but as a result of the feedback gathered during the pilot study (cf. 3.1.5.2), participants were instead only allowed to use the provided corpora (see below) and were specifically asked not to use any other resources (i.e. dictionaries, glossaries or terminology databases such as IATE). Participants were told that the reason for this limitation was that one of the goals of the exercise was to test their ability to find terminology when dictionaries and/or glossaries are either unavailable (because terminology in that area has never been researched) or not usable (because of a hypothetical client's specific requirement that only terminology already present in the official company's documentation be used). Giving this kind of context was considered necessary because of complaints voiced by some participants in the pilot study: since they were accustomed to researching terminology using search engines and online resources such as Word Reference and IATE, they found the

whole exercise pointless and frustrating and felt that forcing them to use only corpora rendered the situation unnatural.

Participants were given access to three corpora:

- FINENOR (~1.4 million tokens) a corpus containing about 50 annual reports of British and North American companies;
- FINITOR (~1.9 million tokens) a corpus containing about 40 annual reports and financial statements of Italian companies;
- FINENTR (~1.7 million tokens) a corpus containing the translations of the texts that form FINITOR.

It should be noted that FINITOR and FINENTR, albeit parallel, were not aligned, so it was not possible to query one corpus and find directly the corresponding translated segment. The corpora were not created specifically for this task, they had been developed some time before within the CONTE project (Gaspari & Bernardini 2010). The corpora were not altered in any way, they were just copied and made available through both systems (Carcass and CQPWeb).

3.1.4 Test subjects

The test subjects were a group of students in their second (and final) year of the master's degree in translation at the Advanced School of Modern Languages for Interpreters and Translators of the University of Bologna.

The experiment was carried out during regular class hours of course "27472 – Specialized translation between English and Italian II". What follows is the "Learning outcomes" section taken from the official course description retrieved from the University of Bologna web site:²

The student:

² <http://goo.gl/Od4CV> (April 24th 2012)

- is to acquire a good working knowledge of the strategies, techniques, traditional and current advanced tools and methods of specialized translation work
- is then able to apply them in the translation of specialized texts of different types and genres within the technical-scientific fields
- is to be conversant with and capable of using the basic techniques of documentation, writing and revising texts, and is able to assess the quality of the texts s/he has produced
- is to be able to identify and apply the advanced translation strategies that are most appropriate to the communicative functions of the source texts

The course was divided into four modules and the experiment was carried out at the beginning of the "Technical/scientific" module, which focussed on

[...] the translation of specialised texts mainly in the domain of economics/finance. When tackling translation and revision in this area, special care will be taken to enhance documentation strategies and the creation of resources to assist translation (translation memories, glossaries, corpora): to this end, CAT and corpus query tools, i.e. IT tools usually used in the profession, will also be practiced³.

As can be seen from these descriptions, students attending the course are expected to have a high level of competence in English. It should also be noted that students are required to pass a highly selective translation test to be admitted to the degree programme and that during their first year they are required to attend (among others) courses in linguistics and specialised translation.⁴ Moreover, since the focus is *specialised* translation, the ability to translate non-specialised texts between Italian and English is also taken for granted. It was therefore assumed that the only linguistic difficulties students would face would be with respect to the unfamiliar language and specialised terminology in the domain of finance and economics.

³ Ibid.

⁴ <http://goo.gl/5HYkE> (March 12, 2012).

Degree	Students
laurea triennale in comunicazione interlinguistica applicata	22
laurea triennale in mediazione interlinguistica applicata	18
laurea triennale in lingue e letterature straniere	14
laurea triennale bilingue in letteratura e civiltà inglese e italiana	1
laurea triennale	1
bachelor of science in foreign service degree	1
a) bachelor in lingue straniere applicate, b) master in traduzione, c) master europeo in traduzione specializzata	1
Total	58

Table 3.1: education

3.1.4.1 Basic demographic information

A total of 70 students were registered for the course but only 61 attended both classes devoted to the experiment. Three of these did not hand in their work at the end of class and were thus excluded from the results. A total of 58 participants were considered valid tests subjects.

In line with the distribution of the School's student population, there was a marked prevalence of female participants in the sample: 49 students (84.48%) were female and 9 (15.52%) were male. The mean age was 25.34 with a median of 25 and a standard deviation of 2.86. Finally, the oldest participant was 43, the youngest was 24.

3.1.4.2 Education

All but one students had a single bachelor's degree, the only exception was a student who also had a "Master's degree in Translation" and a "European Masters in Specialised Translation". All but one students had a background in disciplines related to cultural mediation, linguistics, translation and literature; the only exception had a "bachelor of science in foreign service degree", which indicates the possibility that the

Native language	Number
Italian	56
Flemish	1
French	1
Total	58

Table 3.2: native language of the participants

student had a background in international economics at an academic level.⁵ Table 3.1 shows a summary of the (BA-level) qualifications held by the students when they took part in the experiment.

Given their education, with the possibility of a single exception, it is safe to assume that economics was largely a rather unfamiliar domain for participants and that they could not draw from any solid experience in the field.

3.1.4.3 Linguistic background

The vast majority of participants (56) indicated Italian as their native language, one student indicated French and another Flemish (see table 3.2). Although it was possible to indicate more than one native language, no one chose to do so, hence none of the participants considered themselves to be bilingual.

Given their educational and linguistic background, it can be concluded that most students were working between a language they were very fluent in (English) and their native language (Italian).

3.1.5 Pilot study

The actual experiment was preceded by a pilot study conducted on a small group of people. Three of them (one doctoral candidate and two post-doc researchers) had a degree in specialised translation but did not

⁵ A cursory Internet search of various universities offering this type of degree indicates that students pursuing this degree can major, among other things, in International Economics.

work as translators, one (another doctoral candidate) was a professional translator while the last one (again a doctoral candidate) was a linguist specialising in second language acquisition.

None of the people participating in the pilot study was an expert in corpus linguistic, two of them had used corpora before but not for terminology or documentation. Despite the fact that the participants in the pilot study (with a single exception) had a rather similar profile to the experimental subjects of the experiment, I expected that they would have more problems than students in completing the task because, unlike the students, they had never been trained to use corpora for translation, this turned out to be an accurate prediction (cf. 3.1.5.2).

3.1.5.1 Pilot study description

While the actual experiment had to be divided into two separate sessions (a briefing session and an experimental session) because it had been scheduled to take place during regular class hours (90 minutes long), the pilot was carried out during a single day.

The day started with a briefing session in which the basic features of both Carcass and CQPWeb were presented (i.e. how to log in, navigate the interface, perform a "simple" query, sort results, etc.). After this introduction the rest of the briefing consisted in a hands-on tutorial in which participants were taught how to effectively write CQP queries using part-of-speech tags and lemmas. Once the basics had been covered and participants had an acceptable command of the CQP syntax, there was a short break.

After the break, participants were divided into two groups (designated by the letters A and B). The experiment was divided in four parts, each part was 15 minutes long with a 5-minute break afterwards. In the first part, three terms (in English) were given to participants and their task (detailed in 3.1.2) was to find equivalents and sample sentences of these terms using "Google" (i.e. anything they could find

using a search engine). In part 2 they were given three more terms (again, in English), this time the only resource they could use was one of the two systems being evaluated (group A used Carcass, group B used CQPWeb). In part 3 they were given a new set of three terms, group A was instructed to use CQPWeb, while group B used Carcass. Finally, in part 4 they were given again three terms, but this time they could use anything they wanted: Carcass, CQPWeb or Google. At the end of part 2 and 3, instead of taking a break, participants were asked to respond to a questionnaire (see 3.1.6.3).

Part 1 (Google) was actually skipped because of time limitations on the day the pilot study was carried out: since the goal of the pilot was just to test the validity of the experiment's design, it was assumed Google would be the least problematic system, thus it was deemed acceptable to skip it.

To sum up, the idea was to have participants perform the same task, each time with different terms, for a total of 4 times using: 1) "Google", 2) system A (i.e. Carcass or CQPWeb), 3) system B (i.e. CQPWeb or Carcass) and 4) anything they wanted.

3.1.5.2 Pilot study results

A number of insights were gained from issues that came up in the pilot study. Firstly, participants were either translators not very familiar with corpus-based techniques or they were linguists who had used corpora before but had never worked with terminology. One of them was not a native speaker of Italian and was in the difficult position of working between two foreign languages, of which however he had an excellent knowledge.

Another small problem was of a logistic nature, since they had had little time to learn the CQP syntax participants often needed to go back to their notes to remember how to formulate queries. To solve this problem, in the actual experiment each student was given a one-page

"cheat sheet" containing real examples of CQP queries (see Appendix C) they could use as a quick reference.

The real problem with the participants in the pilot study was not that they had problems writing correct queries but that they had never used corpora to find terminology and did not know how to do it. For instance, some of them insisted on formulating a hypothesis as to what could be the equivalent of a term and kept looking for variations of the term in Italian, instead of looking for known Italian equivalents of collocates of the English term. As a result, most of them did not find even a single equivalent because they spent all their time on the first term. Needless to say this resulted in high levels of frustration.

Finally, another important insight gained from this pilot was that participants did not see the point of the task: why use corpora when you can easily find terminology on the Internet just using Google, Word Reference or IATE? A very good question indeed and one that I tried to answer in 3.1.3. In part 4 (when they could use anything they wanted) all but one participants insisted in using Carcass and CQPWeb: none of them found the equivalent of even a single expression. The one that used Google was able to find all three in under 5 minutes. In this respect it is worth noting that although the expressions proposed were difficult, they could still be found in online specialised dictionaries or terminology databases.

Thanks to the unsatisfactory results of the pilot, it was possible to identify and solve many of the problems that might have emerged during the actual experiment. In particular, one of the most serious doubts I had about the design of the experiment was confirmed: the use of Google introduced a variable that was impossible to control. If students had access to the Internet, than they could find ready-made glossaries and terminology databases which would have rendered the whole exercise meaningless. Even instructing them to use the Google results page as a sort of KWIC page would be (a) difficult to enforce and

(b) pointless because the result pages often contains snippets of texts taken from glossaries. On a more general note, it was felt that the use of Google would be a distraction from the main goals of the exercise, which was to evaluate the users' perception of the two corpus query systems being investigated and to observe the way they interact with corpora. It was thus decided to eliminate part 1.

Part 4 was also eliminated and replaced with a questionnaire (questionnaire 3) in which participants were specifically asked to give their opinion and compare the the two systems.

As to the objection that forcing a translator to use only corpora was unrealistic, I tried to address it by drawing the attention of students to the fact that dictionaries and terminology databases are not always reliable and very often incomplete, especially when dealing with technical texts: anyone can use a dictionary, but what happens when you do not find the term you are looking for? A translator should be able to use primary sources (i.e. real documents in the same domain as the text being translated) to find terminology, a corpus is just a more powerful tool to search texts.

I also emphasised the fact that, although they were working with the Italian-English language combination, the same methods could be applied to other language combinations for which the availability of pre-made dictionaries and terminology databases might be more limited.

3.1.6 The experiment

Since, unlike the pilot, the actual experiment was carried out during regular class hours (90 minutes) it was necessary to divide it in two separate sessions: a "briefing" class, in which students would learn to use the tools needed to complete the task, and a "task" class (held a week later) where the actual experiment would take place and results would be recorded.

The time interval between the two classes had two advantages over the pilot: 1) the sessions were shorter and thus less intensive and 2) participants had a chance to practise at home with the two systems, in fact they were encouraged to do so (although it was not possible to determine how many students used the corpora during the week, server logs show that at least some of them did).

3.1.6.1 Class 1 (briefing)

During the first class students were told that the university was in the process of acquiring a new Corpus Management Tool and was trying to evaluate which one was better suited to students' needs. The choice had been narrowed down to two systems: Carcass and CQPWeb. Participants were not told that Carcass had been developed internally and that the true purpose of the task was to evaluate their reaction to it in comparison with a baseline.

The class lasted 90 minutes and was held in a computer lab where each student could use a computer. All students had already used corpora during their first year, were familiar with the concepts of concordances and collocations and were proficient in the use of AntConc (Anthony 2005),⁶ while none of them knew the CQP syntax.

After a brief introduction in which the basic functionalities of both systems were presented (i.e. how to log in, how to perform a "Simple Query", how to sort results, etc.), most of the class was spent in a hands-on session in which students were taught how to effectively write CQP queries using part-of-speech tags and lemmas. Both systems were used to illustrate examples and students were encouraged to try to use both systems equally.

Towards the end of the lesson, the task students would be assigned the next week was simulated: they were given three expressions in English and were asked to devise corpus-based strategies to find the

6 <http://goo.gl/3GVS>

	Group A	Group B	Group C	Group D
Part 1	Carcass CQPWeb Expression set 0	Carcass CQPWeb Expression set 0	CQPWeb Carcass Expression set 0	CQPWeb Carcass Expression set 0
Part 2	CQPWeb Expression set 1 Questionnaire 1	CQPWeb Expression set 2 Questionnaire 1	Carcass Expression set 1 Questionnaire 1	Carcass Expression set 2 Questionnaire 1
Part 3	Carcass Expression set 2 Questionnaire 2 Questionnaire 3	Carcass Expression set 1 Questionnaire 2 Questionnaire 3	CQPWeb Expression set 2 Questionnaire 2 Questionnaire 3	CQPWeb Expression set 1 Questionnaire 2 Questionnaire 3

Figure 3.1: set-up of the experiment

equivalent expressions in Italian. During this exercise session they could discuss their ideas in groups or with teachers, successful strategies were then shared with the rest of the class and commented by the lecturer of the course.

3.1.6.2 Class 2 (task)

At the beginning of the second class (a week after the first), participants were divided into four groups (named A, B, C and D) and arranged in such a way that members of the same group were separated by 3 other students (see Figure 3.1). Each participant would sit next to a maximum of two people: one would be using the same system but working on a different set of expressions, the other would be using a different system and working on a different set of expressions. This arrangement was devised to minimise the chance of students copying and to avoid a bias in the collected data due to a learning curve effect (i.e. students might feel more comfortable using the tools towards the end of the exercise, after having familiarised with the CQP syntax).

All participants were given a handout containing:

- a printed instruction sheet detailing every step of the experiment, instruction sheets were different for each group and had been customised for each participant with a unique username and password to access CQPWeb (cf. Appendix A);
- a printed four-page extract from "EasyJet Annual report and accounts 2010" containing the expressions they were asked to investigate; this was meant to be a quick reference in case they needed more context to understand the English expressions (which had been highlighted in the text, cf. Appendix B);
- a printed "cheat sheet" containing a reminder of the basic CQP syntax and a list of common pitfalls (cf. Appendix C);
- a table (in electronic format) containing the tagsets of the three corpora (cf. Appendix D);
- a "Results table" (in electronic format) that students were asked to fill out with the results of their terminology research (cf. Appendix E);
- the full "EasyJet Annual report and accounts 2010" in electronic format.⁷

The task was explained once more and participants were again reminded that they were not permitted to use dictionaries or other online resources. Since objections to these limitations had been raised by participants in the pilot (see 3.1.5.2) the reasons behind them (outlined in 3.1.3) were made abundantly clear in order to prevent complaints from students who might consider the whole exercise pointless.

Part 1 (warm-up)

After the introduction, the experiment started with a 10-minute warm-up in which groups A and B were asked to use Carcass to find the Italian

⁷ The complete text is available online at <http://goo.gl/eA0WI> (April 24th 2012)

equivalent of the expression "functional currency". They were then asked to use CQPWeb to find an example of the Italian equivalent. Groups C and D were asked to do the same using CQPWeb to find the Italian equivalent and Carcass to find the example.

In the warm-up both systems were employed for the same expression to make sure that each of them had been used at least once before the beginning of the experiment. During the task, students were instructed to use only one system at a time. Results from the warm-up phase were recorded but discarded from the analysis.

Part 2

After a short break, the actual task started, the time limit for this part was 15 minutes.

Groups A and B were instructed to use only CQPWeb, groups C and D were told to use Carcass. Groups A and C had to investigate the expressions "assets given", "diluted earnings per share" and "intangible assets". Group B and D investigated "under operating leases", "net book value" and "hedging instrument".

Participants were encouraged to work for no more than five minutes on each expression, if they could not find a solution in five minutes they were advised to move on to the next one. This was just a suggestion, students were left free to choose how to manage their own time. At the end of this part, participants were given 5 minutes to fill out Questionnaire 1 (see 3.1.6.3).

Part 3

The time limit for this part was 15 minutes. Again, participants were encouraged to work for no more than five minutes on each expression.

Groups A and B were instructed to use only Carcass, groups C and D were told to use CQPWeb.

Groups A and C had to investigate the expressions "under operating leases", "net book value" and "hedging instrument". Group B and D investigated "assets given", "diluted earnings per share" and "intangible assets".

At the end of the allotted time, participants submitted their work and were given 5 minutes to fill out Questionnaire 2 (see below) and then 10 more minutes to fill out Questionnaire 3 (see below).

3.1.6.3 Questionnaires 1 and 2

The main purpose of this experiment was to test whether Carcass would be an acceptable alternative to a more traditional, web-based corpus query system (i.e. CQPWeb). The widely used System Usability Scale (Brooke 1996: 189), or SUS, was employed to assess the students' perception of the usability of the system. The SUS is a Likert scale, consisting of 10 statements to which students had to indicate their degree of agreement on a 5-point scale.

The SUS questionnaire was administered twice, at the end of part 2 and then again at the end of part 3 (in the experiment outline provided above I called them "Questionnaire 1" and "Questionnaire 2"). Both questionnaires were identical and students were asked to evaluate only the system they had just used, immediately after having finished using it.

The statements included in the SUS are rather unspecific as to the type of system that is being evaluated, participants are asked to respond to statements such as "I think that I would like to use this system frequently", "I would imagine that most people would learn to use this system very quickly" and "I felt very confident using the system" (the full questionnaire is available in Appendix F).⁸ The general tone of these statements and the lack of focus on specific features are the main reasons why the SUS was considered ideal for the evaluation task

⁸ All questionnaires were in Italian.

proposed here: since the party proposing this comparison has an obvious interest in "pushing" one of the two systems, it was considered best to use a standardised method for comparing the two software applications, as opposed to devising a set of questions which might have been biased, either in their formulation or in the choice of aspects to evaluate (a – possibly biased – questionnaire focussing on a distinct set of feature was proposed at the very end of the experiment as "Questionnaire 3", see 3.1.6.4)

Another reason for choosing the SUS was that it produces a usability score (a number between 0 and 100) that can be very useful in comparing different systems. In his introduction to the System Usability Scale, Brooke (1996, p. 190) argues that:

often, all that is needed is a general indication of the overall level of usability of a system compared to its competitors or its predecessors. Equally, when selecting metrics, it is often desirable to have measures which do not require vast effort and expense to collect and analyse data.

By comparing the score assigned by the user to system A with the score assigned to system B it is possible to obtain a clear indication of which one the user preferred without explicitly asking (in Questionnaire 3 the question "which system do you prefer" was asked explicitly, but this came only later).

The SUS is scored by assigning a value to the response to each statement, which range from 1 (strongly disagree) to 5 (strongly agree). Odd numbered statements (which are favourable to the system) are assigned a score equal to the scale position minus 1, even numbered statements (giving a negative assessment of the system) are assigned a score of 5 minus the scale position. To obtain the final value, all scores are added together and multiplied by 2.5. The range of the SUS score is 0 to 100.

It should be noted that all questionnaires were administered using a web-based tool⁹ which allowed statements to be presented in a random order, different for every user, a strategy that should have prevented participants from answering in a pattern.

By administering the questionnaire immediately after each of the two systems had been used, it was assumed that the usability score would not be biased towards one of the systems and would be as objective as possible (the obvious caveat of course is that since the evaluation was based on user perception it is inherently subjective).

3.1.6.4 Questionnaire 3

Unlike questionnaires 1 and 2, which were used to indirectly determine the user preference for one of the two systems, questionnaire 3 asked users to explicitly state their opinions. In the first part, participants were presented with a set of 5-point scales in which Carcass was at one end and CQPWeb was at the other. They had to give their opinion on which system came closer to the statement being proposed. The statements all began with "State in which one of the two systems" (the following list was then presented in random order):

1. it is easier to switch between corpora
2. results visualisation is clearer
3. it is more difficult to find functionalities
4. sort is easier to use
5. functions are easier to find
6. it is easier to formulate a query
7. the query history is easier to use
8. fewer accidents occur

⁹ I used *Limesurvey*, <http://www.limesurvey.org>

Participants were then asked to explicitly state up to three aspects that, in their opinion, made one system better than the other. They were also encouraged to explain the reasons why they thought so.

In the next part students were asked which systems they found easier to use (Carcass, CQPWeb or both), which one they preferred (again they could choose Carcass, CQPWeb or both) and they were also given the option to leave a comment.

The following question was "if in the future you had to perform a similar translation/documentation task, which of the two systems would you use?" They were given the possibility to express their opinion as "mostly Carcass", "mostly CQPWeb", "both" or "neither".

The final question was "which of the two systems would you recommend to a colleague (student or translator) who had to use corpora for a translation?", here the options were "Carcass", "CQPWeb" or "both".

CHAPTER 4

RESULTS AND DISCUSSION

In this chapter I will present the results of the experiment described in chapter 3. I will start by analysing the SUS scores derived from Questionnaires 1 and 2, then I will present the opinions expressed by users in Questionnaire 3. In the final part of the chapter I will analyse the user behaviour that emerged from inspection of the query logs, offering a general quantitative overview and a progressively more in-depth examination of two subsets of users (one group is formed by the 5 worst-scoring student, the other by the 5 best-scoring students).

4.1 SUS score

As discussed in 3.1.6.3, Questionnaires 1 and 2 were used to obtain an independent SUS score, ranging from 0 to 100, for the two systems. Carcass obtained a mean score of 60.30, with a standard deviation of 17.79 and a median of 60.00. CQPWeb obtained a mean score of 58.92, with a standard deviation of 19.34 and a median of 62.50 (Figure 4.1 shows mean and median scores of the two systems, while Figure 4.2 shows a comparison of the scores).

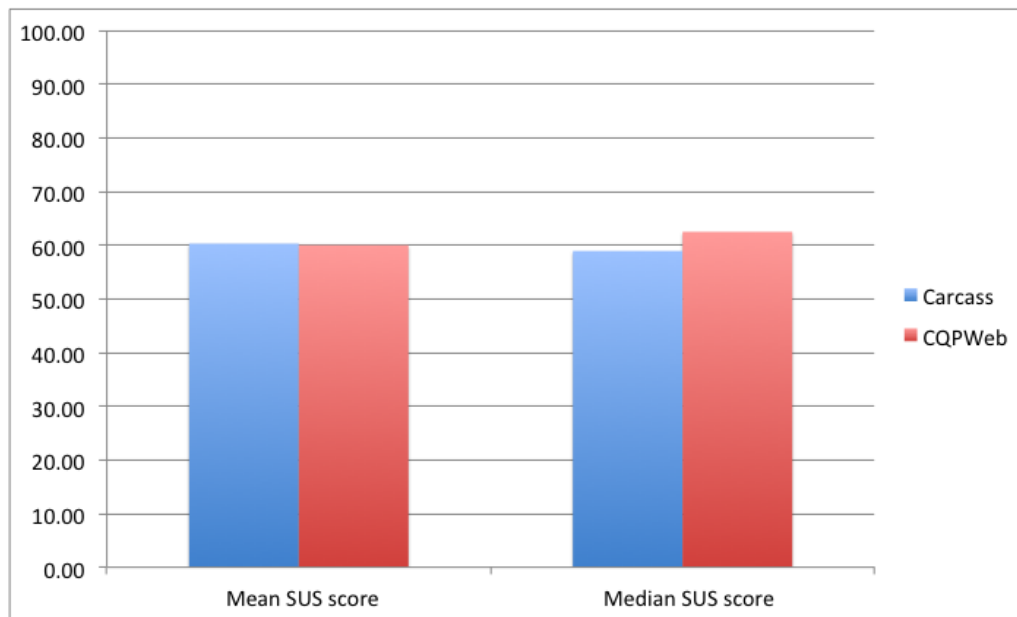


Figure 4.1: mean and median SUS scores for Carcass and CQPWeb.

The two system obtained a very similar score with an equally high standard deviation, which indicates that most participants found both systems to be either particularly easy or particularly hard to use. It should be noted that while every effort was made to make sure that the formulation clearly indicated that it was the *software application* that

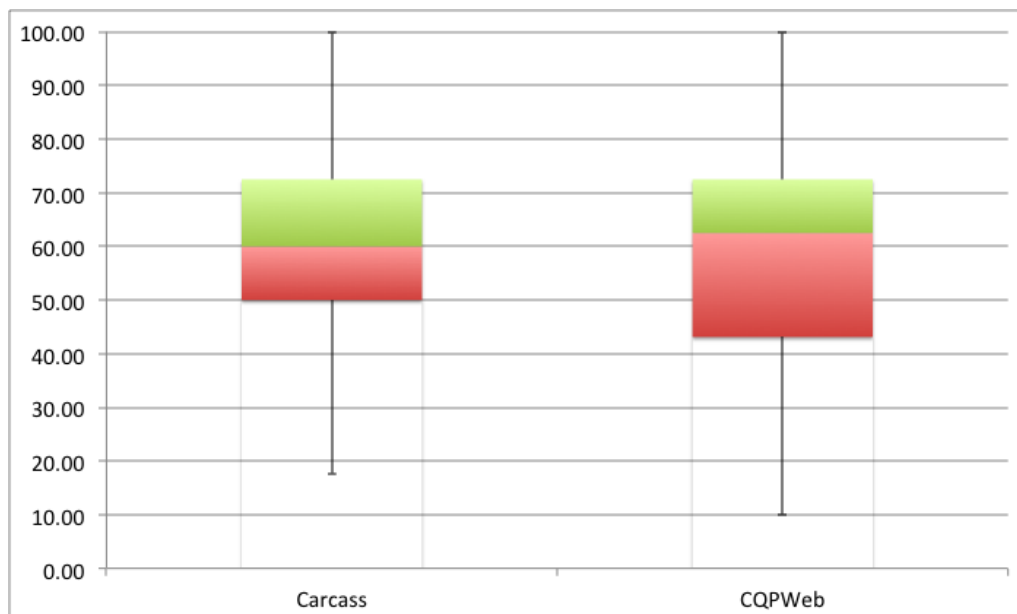


Figure 4.2: distributions of the SUS score for Carcass and CQPWeb.

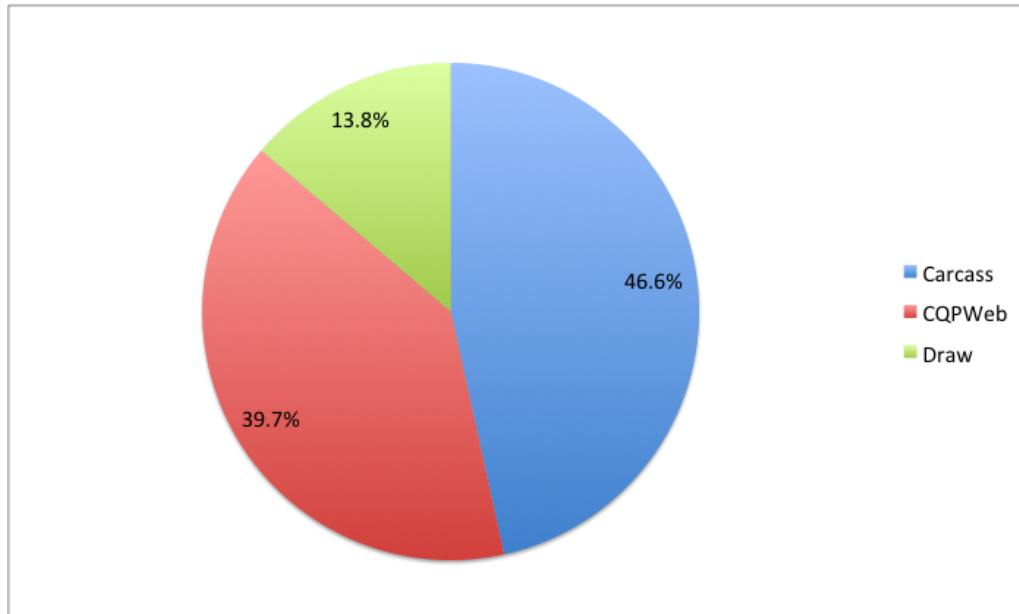


Figure 4.3: "favourite" system as resulting from the comparison between the SUS score assigned by each participant.

was being evaluated, it is entirely possible that the inherent difficulty of the CQP syntax had an impact on the evaluation. But even if this were true, it would not affect the validity of the comparison since both systems use the CQP syntax for their "advanced query" functionality.

As the two systems had very similar scores, in an attempt to determine whether there was indeed a preference for one of the two, the SUS scores assigned to the two systems by each participant were compared and the higher scoring system was marked as "favourite". From this comparison, it emerged that 46.6% of participants gave a higher score to Carcass while 39.7% gave it to CQPWeb, the remaining 13.8% of users attributed the same score to both systems (see Figure 4.3).

The data show that the two systems were perceived as being equally usable by users, with only a slight preference for Carcass emerging from the direct comparison of the individual scores.

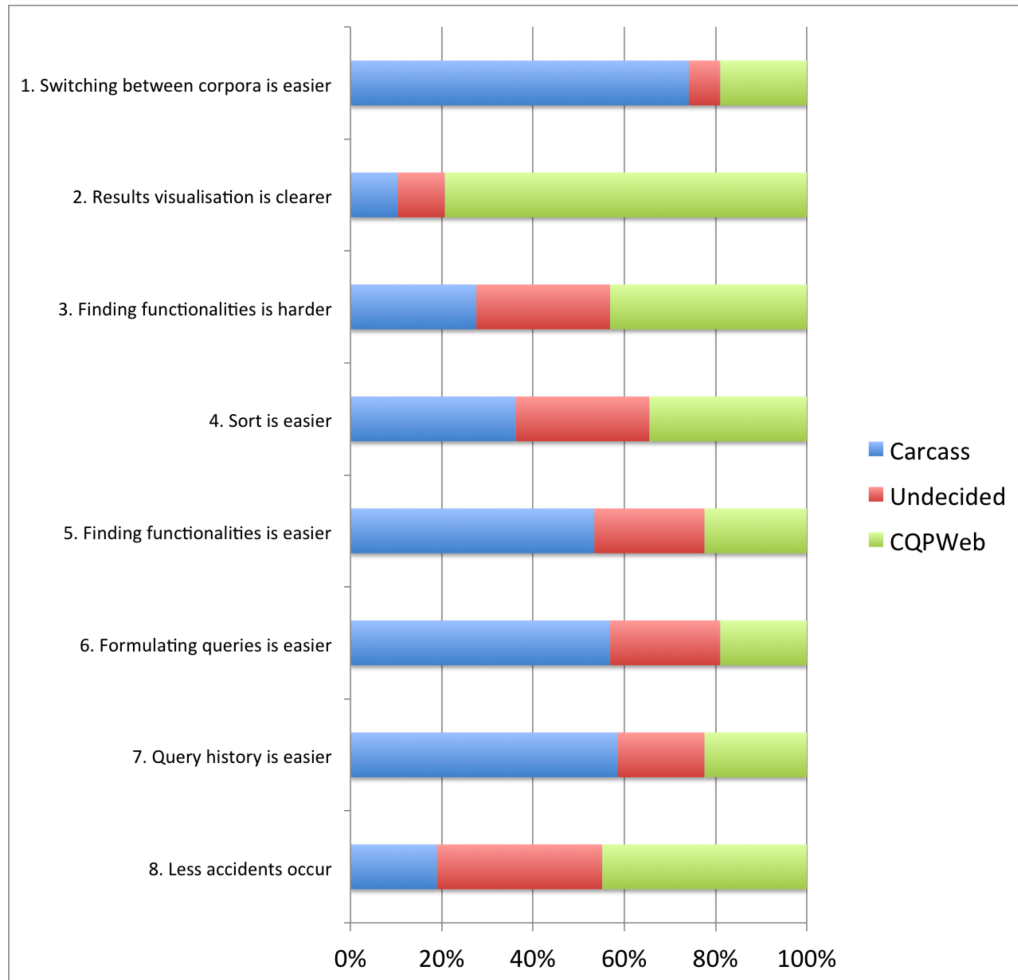


Figure 4.4: comparison of a subset of features of Carcass and CQPWeb

4.2 Users' opinions

At the end of the task, participants were required to answer a third set of questions (Questionnaire 3, see 3.1.6.4) designed to gather participants' opinion on the two systems. Some of the questions also offered the possibility to add comments which proved to be very helpful in the interpretation of the results (Appendix G contains a full transcript of the comments left by participants).

4.2.1 Feature comparison

The first set of statements was aimed at comparing the way Carcass and CQPWeb implement a specific set of features. The comparison does not encompass all available features in both systems, in fact it is limited to

Feature	Number of "votes"	
	Carcass	CQPWeb
Cleaner interface	2	1
Simple search	1	
Smart search	7	
Corpus selector	3	
Single window	1	
Query history	2	1
Show full text		5
Stability		1
Results visualization		6

Table 4.1: users opinions on what made one system better than the other

the main characteristics implemented by Carcass since it was designed to test whether the effort put in a few specific areas of the interface would indeed be appreciated by users. In presenting these results I am not suggesting that one system is better than the other, I will only argue that most users think that the very specific set of features examined here are better implemented in one of the two. It will also become apparent that under more than one point of view, the two systems were considered virtually equivalent.

While the comparison of the SUS scores (reported in 4.1) suggests that the users did not think one system was clearly better than the other, the data collected in questionnaire 3 show a marked preference for one of the two systems at least in some respects, Figure 4.4 summarises the results.

Users seemed to appreciate the fact that Carcass offers them the possibility of quickly switching from corpus to corpus (statement 1 of Figure 4.4), this is made possible by the "Corpus selector" window that constantly displays all available corpora. It should be noted that even though it was expected that the task would require users to switch

frequently from the English corpus to the Italian one, a close examination of the query logs (see 4.3) showed that few users did actually do that. Nevertheless the comments left by participants (see below and table 4.1 for more information on user comments) suggest that the feature was appreciated.

CQPWeb's full screen result visualisation proved to be much more appreciated than the small results window offered by Carcass (statement 2). The erratic behaviour of the display window, which tends to unexpectedly scroll left or right to accommodate lines of text, was probably another source of confusion. Also Carcass does not offer the possibility of displaying the full source text for a concordance, a feature that, according to user comments, was much appreciated in CQPWeb.

Moving on to statements 3 and 5 (which basically asked the same thing, one said "state in which one of the two systems it is more difficult to find functionalities" the other "state in which one of the two systems functions are easier to find"), the data show that participants thought that Carcass made it easier for them to find what they needed. It is worth noting that Carcass has a much more reduced set of features than CQPWeb, this could be a factor because fewer features result in a "leaner" interface that is easier to navigate.

There is no clear indication of a preference between the implementations of the "sort" function (statement 5), while there is evidence (also supported by comments, cf. Appendix G) that users liked the three different editors available in Carcass and especially welcomed the "Smart query" mode which allowed them to save time when formulating complex queries (cf. 4.3).

Finally, participants in the experiment preferred the query history implemented in Carcass (statement 7) while they thought that CQPWeb was less prone to accidents (statement 8). In this respect it must be noted that several users had to restart Carcass because of a crash, an

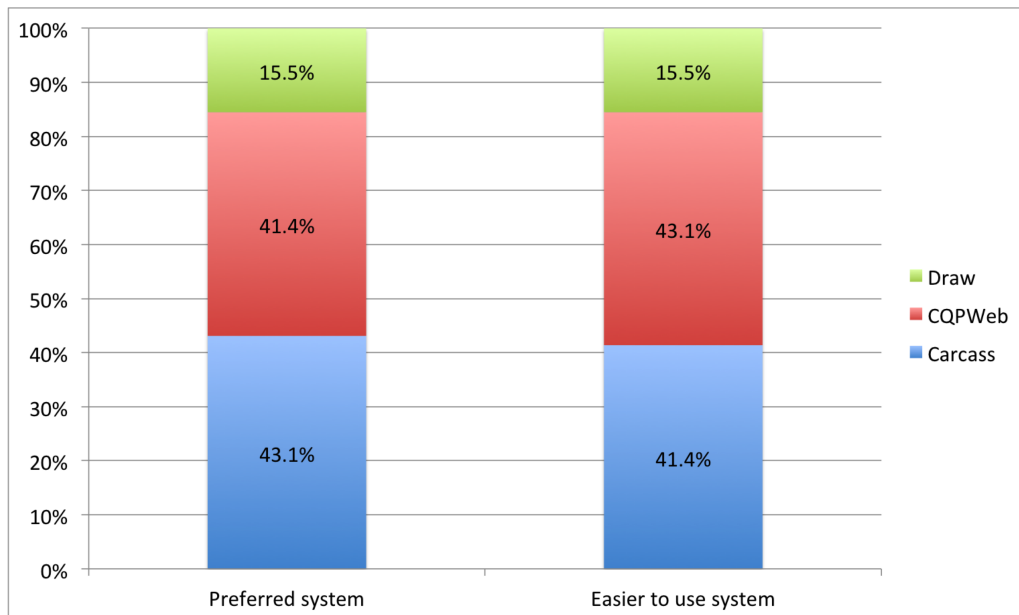


Figure 4.5: users opinion on which system they preferred and which one they thought was easier to use

occurrence that had been largely expected given the immaturity of the application.

Users were also given the option to indicate up to three aspects that, in their opinion, made one system better than the other. Table 4.1 shows a summary of the results: the first column lists the aspects that were mentioned by participants while columns 2 and 3 show the number of users who manifested their preference for that particular aspect.¹

These comments confirm what transpires from the answers to Questionnaire 3 and clearly indicate that Carcass needs to become more stable and must improve result visualization. A "show full text" functionality would also be very appreciated by users.

4.2.2 Overall system comparison

Two of the questions in questionnaire 3 ("which system do you prefer?" and "which system did you find easier to use", see 3.1.6.4) were

¹ The full list of comments is available in Appendix G.

intended to confirm the data gathered in questionnaire 1 and 2. Although they are not the same as "which system is more usable?" (which is the question underlying the comparison between the scores obtained from questionnaires 1 and 2), it was assumed that the two could effectively subsume the concept of usability, without having to explain to participants what was meant by usability.

To the first question, "which system do you prefer, Carcass or CQPWeb", 43.1% of participants said they preferred Carcass, 41.4% stated they preferred CQPWeb while the remaining 15.5% said they liked them both equally. The answers to the second question, "which system did you find easier to use" were in line with those from the first one: 41.4% said Carcass was easier, 43.1% stated CQPWeb was easier while again 15.5% said there was no difference between the two (see Figure 4.5).

These figures are in line with the results presented in 4.1 where it was shown that there was no clear preference for one system or the other.

4.2.3 Users intentions for future use

The next question was "if in the future you had to perform a similar translation and/or documentation task, which of the two systems would you use?". The results were again inconclusive: 43.1% said they would use Carcass, 39.7% would use CQPWeb, while 8.6% would use both and another 8.6% would use neither system.

An entirely different result came out of the next question, where users were asked which system they would recommend to colleagues: 24.1% said they would recommend Carcass, another 24.1% would recommend CQPWeb and 51.7% would recommend both systems. These figures might well be an artefact since in this case there was no option to say "neither", but they can also be explained by the fact that only a

fraction of the users felt strongly for one system or the other, while most of them considered them basically interchangeable.

The data gathered in questionnaire 3 corroborates the results of questionnaires 1 and 2: there is no clear winner in the comparison. The data contain nonetheless very useful indications of areas where the two systems can be improved: the way Carcass displays results needs a major overhaul and the possibility to present the full text of a concordance must be added. The frequent crashes reported by users are a strong indication that the whole Sarcophagus system in general needs to become much more stable. On the other hand, CQPWeb could benefit from a cleaner, more integrated interface and a simplified query editor such as Carcass's "Smart search". Improvements on the handling of the query history and a simpler way of changing the active corpus would probably be appreciated by users.

The results are also very encouraging for the new project Sarcophagus: despite its immaturity nearly half the participants preferred Carcass over CQPWeb, a much more stable and mature system.

4.3 Analysis of users behaviour

In this section I will examine the users' behaviour that emerged from the analysis of the query logs of the two systems.

The experiment was completely anonymous but participants were asked to provide the CQPWeb username and the name of the machine they were using at the beginning of each questionnaire and on the "results table" they handed in at the end of the task. Taking advantage of server logs, I used these data to match users to the queries they submitted and I was able to reconstruct the full query history of all users on both Carcass and CQPWeb.

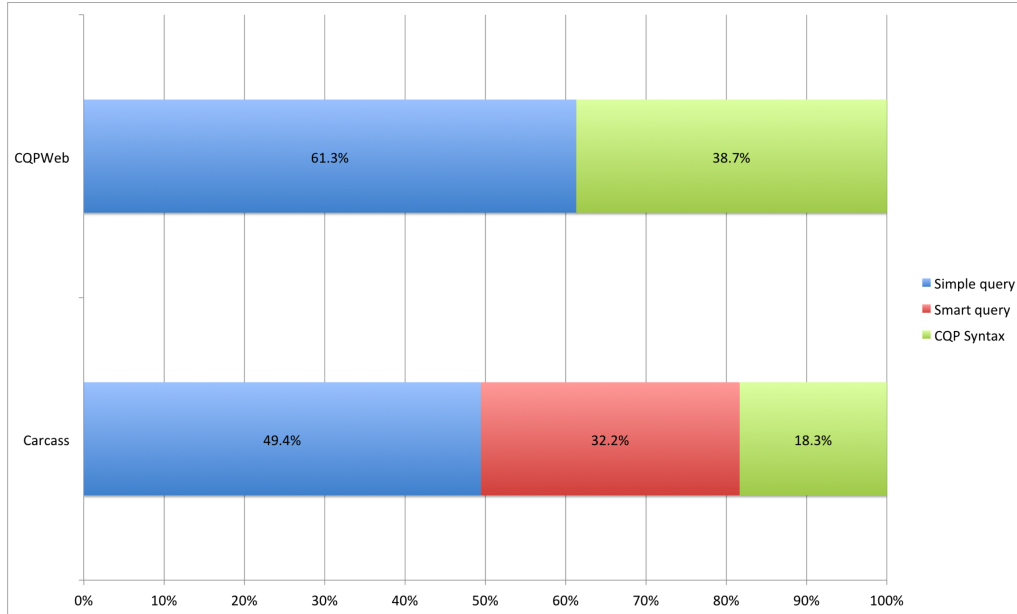


Figure 4.6: breakdown of queries according to query mode.

4.3.1 Query modes

The total number of queries submitted by participants via Carcass during the experiment was 1,343 while the total for CQPWeb was 1,270. Each user submitted an average of 23.15 queries on Carcass and 21.89 on CQPWeb, for a total mean of 45.05 queries during the whole experiment.

Again there is a remarkable similarity in that the total number of queries is very similar, although differences do emerge when examining the "query mode" (i.e. simple, smart and CQP syntax) employed by users. Figure 4.6 shows a breakdown of queries according to query mode which reveals that a clear majority of queries submitted via CQPWeb were typed in the "simple" editor, whereas less than half the queries submitted in Carcass were "simple". A more interesting consideration that can be made looking at the data though, is that the portion of "CQP syntax" queries submitted via Carcass (18.3%) is much smaller than those submitted via CQPWeb (38.7%) and that this seems to be compensated by a widespread use of the "smart query", which accounts for 32.2% of all queries submitted by Carcass users. If we

consider that "smart" and "CQP syntax" queries can be used to express sophisticated patterns, well beyond the possibility offered by the simple mode, it is interesting to observe that slightly more than half (50.5%) of the queries submitted by Carcass users can be considered "advanced" while CQPWeb users used the CQP syntax only for 38.7% of their queries.

A very likely explanation for this is that, in most cases, Carcass's "smart" query (which is exclusive to this system, CQPWeb has nothing like it) can be an effective substitute for the CQP syntax. It also appears that the possibility of using the "smart editor" is the reason why simple queries are less popular with Carcass users than with CQPWeb users: instead of trying a simple query, users felt that the smart editor allowed them to quickly formulate slightly more complex queries without having to resort to the CQP syntax. This was confirmed by several users who commented on how, thanks to the smart query, they did not "have to remember" the CQP syntax (see 4.2.1 and Appendix G).

Another constant that emerged from a more in-depth analysis of the query logs is that once users try a more advanced mode they have a tendency to continue using it, especially when refining queries (cf. 4.3.3), rather than going back to using the simple query. This is probably because it is easier to just add, remove or edit small portions of text rather than to start from scratch using a different syntax.

In conclusion, participants appreciated the availability of different query modes and used all of them, albeit with varying degrees of success in the more advanced modes (cf. 4.3.2). They especially welcomed the smart query editor as a convenient replacement for both the simple and the advanced query modes, a way of searching sophisticated patterns without having to go through the more lengthy process of typing a full CQP query.

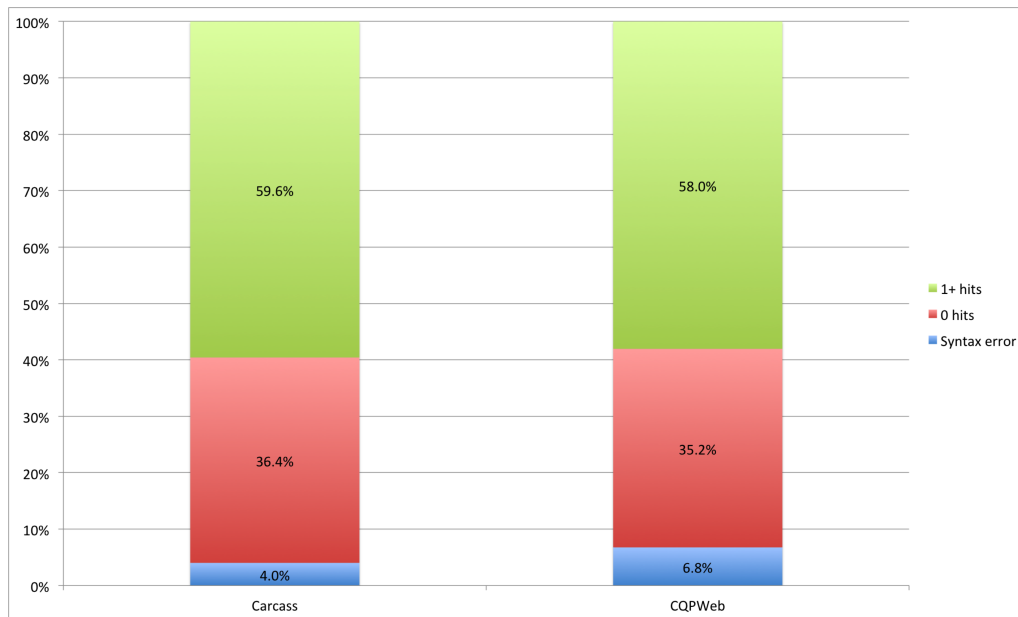


Figure 4.7: query results

4.3.2 Query results

Another important consideration that can be made by analysing the query logs is with respect to the number of hits returned by queries. Figure 4.7 shows how many queries produced a syntax error, how many were formally correct but returned zero matches and how many returned 1 or more matches. The percentages of the last two categories are very similar in both systems (but see 4.3.2.2), whereas there appear to be a significant difference in the number of syntax errors: 6.8% of queries submitted to CQPWeb were formally incorrect, while only 4.0% of those submitted via Carcass were syntactically wrong.

In a similar study, Santos and Frankenberg-Garcia (2007: 350) claim that server logs for their *Compara* corpus show that "queries that produced zero hits consistently amount to roughly half the total number of queries" (they count syntax errors as o-hits queries) whereas the data reported in Figure 4.7 show that in our case only about 40% of queries are either errors or o-hits queries. This difference could be due to the fact that *Compara* is a web-based tool that might attract a certain number of casual users (and possibly even some robots that could

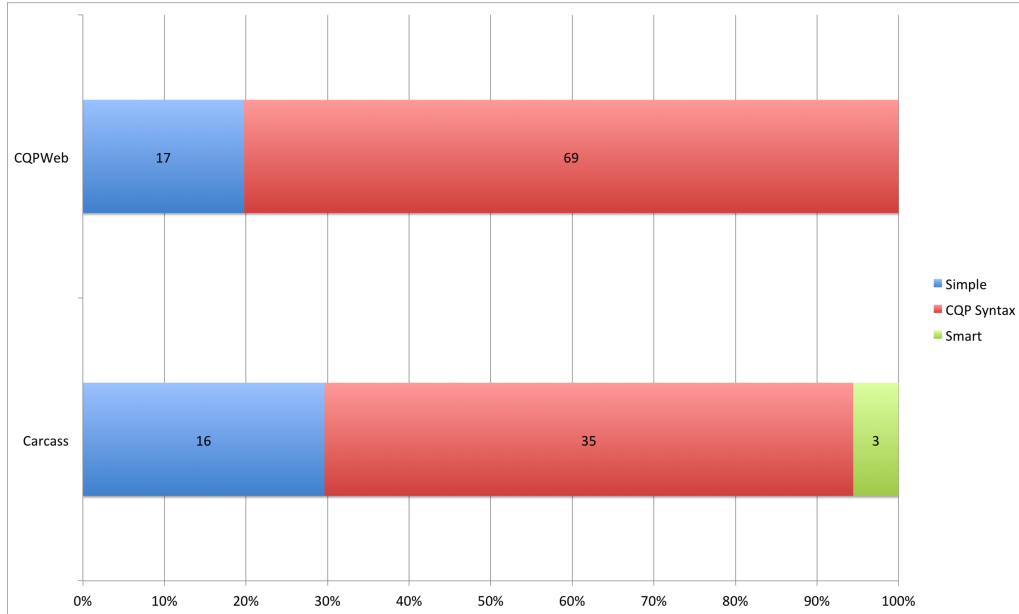


Figure 4.8: distribution of errors according to query mode

simulate queries in order to crawl the site) while the experimental subjects were highly motivated and somewhat trained in the use of corpora.

4.3.2.1 Syntax errors

Figure 4.8 shows a breakdown of errors according to query mode. Almost the same number of errors (16 vs. 17, or 1.2% vs. 1.3% of all queries) were made in the simple query mode and an analysis of the query logs shows that in both tools all these errors stemmed from the fact that the CQP syntax was used in the wrong query mode: when using the CQP syntax in CQPWeb, users are required to select the appropriate option in a drop-down menu while in Carcass they have to type the query in the "Expert editor" window. Using the CQP syntax when the program does not expect it typically results in a syntax error in both systems.

As it was expected, the number of errors increased when the "expert" query mode was used, in this case errors were caused by missing quotes, whitespace or square brackets, spelling mistakes (i.e.

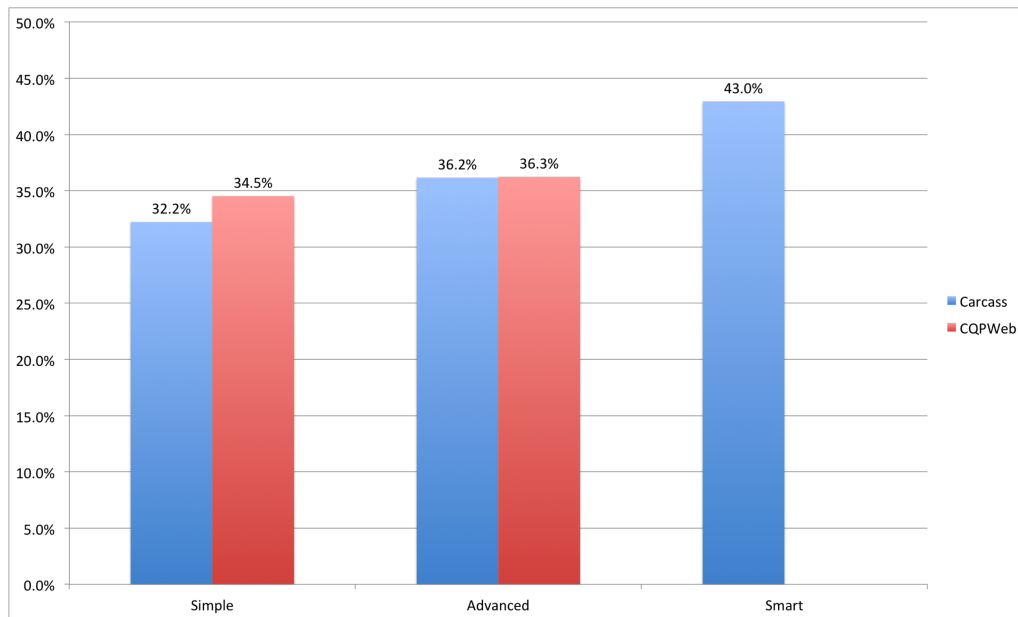


Figure 4.9: percentage of 0-hits queries according to query mode

POS in uppercase, "lem" instead of "lemma" or vice versa), the use of a colon instead of the equals sign, and the wrong use of the Kleene star operator (i.e. the "asterisk"). There were a total of 35 errors in Carcass and of 69 in CQPWeb (2.6% vs. 5.4% of all queries), this difference can be explained by the fact that fewer queries were submitted using the expert mode in Carcass (246 vs. 491) since users of this last system preferred on many occasions the more friendly "smart query". The error rate for the CQP syntax is basically the same in both tools: 14.2% of all advanced queries submitted to Carcass resulted in errors against 14% for CQPWeb.

Only 3 errors were made by users of the smart query editor (whose error rate is 0.7%): one was caused by submitting an empty form (probably the user clicked on the submit button by mistake), the other two were caused by the wrong use of the Kleene star operator.

The data show that neither tool is better than the other in helping the user avoid mistakes when directly typing queries in either the simple or CQP mode, while the "smart" editor appears to make a difference: in many cases users opted for it and were able to search for sophisticated

patterns without incurring in as many syntax errors as they did when using the expert query editor. The downside of this is that, all too often, these queries returned no results (see below).

4.3.2.2 *0-hits queries*

Despite the fact that both systems return a similar percentage of 0-hits queries, a breakdown of the types of queries that produced zero results offers an interesting perspective on user behaviour.

Figure 4.9 shows the proportion of queries that returned an empty result set for each query mode. It comes as no surprise that simple and advanced queries behave in a similar way across the two systems but it is interesting to observe that 43% of all queries submitted using the smart editor returned no results. A close analysis of a subset of the query logs (cf. 4.3.3) reveals that on many occasions users tended to compose overly complicated if not outright byzantine queries which often resulted in patterns that did not make much sense. For example the 0-hits query

```
(a) [word="attività" %cd][lem="dat.*" & pos="VER:ppast"][pos="PRE"]2
```

is overly restrictive: the use of "dat.*" seems to indicate that what the user hoped to obtain was an expression like "attività date" (which incidentally would have been a partially correct equivalent of "assets given", see below). But since there was a restriction on the part-of-speech that forced that particular token to be a verb, there is no possibility of obtaining any results because the lemma of "date" as a verb would be "dare".³ The correct query to find the pattern would be

```
(b) [word="attività" %cd][lem="dare" & pos="VER:ppast"][pos="PRE"]
```

2 This pattern returns all occurrences of the word "attività" followed by a word whose lemma begins with "dat" and whose tense is the past participle, followed by a preposition

3 "date" could also be an adjective but in that case the lemma would be "dato" and it would indeed be captured by "dat.*".

but again it would be a bit redundant because the part that requires the lemma "dare" to be in the past participle could just as well be expressed using a more simple constraint on the word, i.e.

```
(c) [word="attività" %cd] [word="dat.*" %cd]
```

Another example of an overly complicated query is this:

```
(d) [word="data" %cd & pos="NOUN"] [word="di" %cd & pos="PRE"]  
[word="cambio" %cd & pos="NOUN"]
```

there was no need to specify that "di" must be a preposition since it could not be anything else, and there was also little need to specify that "data" and "cambio" had to be nouns since the probabilities that they would have different POS-tags in the context of that phrase were very slim (in fact there were no occurrences of the expression in the corpus, even without the constraints on the part-of-speech).

This overuse of POS-tags constraints was very frequent and in many cases it seems to indicate that users tried to replicate the exact structure of the expression in the source language. In other cases it seemed they were trying, often without much success, to restrict particularly large result sets to a more manageable size.

Another common mistake that does not result in a syntax error but that generally results in a 0-hit query is the misuse of the Kleene star operator (i.e. the asterisk). Users often used it without the "dot" that normally precedes it to indicate "any character"; for instance it is fair to assume that in a query such as this one

```
(e) [word="ricav*"] [pos="ADJ"]
```

the student meant to look for a word that started with "ricav" and ended with any combination of characters (a pattern that should have been expressed using "ricav.*", note the "." before the asterisk) and not a word that starts with "rica" and ends with zero or more letters "v", which is what the pattern in (e) expresses.

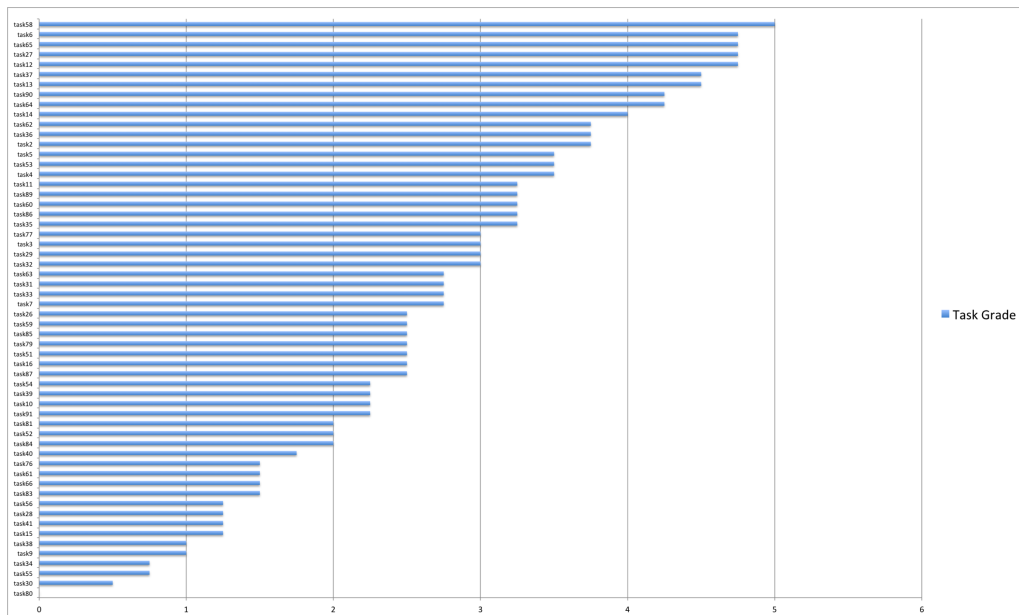


Figure 4.10: task grades

In conclusion, o-hits queries were sometimes caused by subtle mistakes like a wrong or misspelled POS-tag, by the misuse of the Kleene star, but very often also by an excess of enthusiasm in the use of unnecessary part-of-speech or lemma constraints.

4.3.3 Student performance in the task

After having drawn a few conclusions on the general behaviour of the sample population, in this section I will offer an overview of the performance of students with respect to the task they were asked to complete. Then I will examine in some detail the behaviour of two subsets of users, the five worst scoring students and the five best scoring students⁴, in an attempt to find out if conclusions can be made as to which corpus-based strategies have a greater chance of success in this kind of task.

4.3.3.1 Overall student performance

Users scores were assigned by grading the results they handed in at the end of the task (cf. 3.1.6.2). A master list of all the Italian equivalents

⁴ The total population was 58 (cf. 3.1.4.1).

found by users was compiled and then the lecturer of the course was asked to mark each of them as "correct", "partially correct" or "wrong". Partially correct equivalents are those where only a part of the expression was correctly translated, for example the correct Italian equivalent of "net book value" is "valore netto contabile", thus "valore netto" was considered partially correct. Users were awarded 1 point for each correct answer, $\frac{1}{2}$ point for partially correct ones and $\frac{1}{4}$ point for wrong ones, zero points were awarded when no answer had been given. The maximum total score possible was 6, the minimum was 0. No students achieved a perfect score (the best score was 5) while there was a single 0. Figure 4.10 shows the grades assigned to students, the mean of these grades is 2.7 with a standard deviation of 1.2.

An attempt was made to determine whether there was a correlation between the number of correct / partially correct / wrong equivalents and the system used: no significant difference emerged, apart from a slightly higher number of correct answers found by Carcass users and a slightly higher number of partially correct answers found by CQPWeb users.

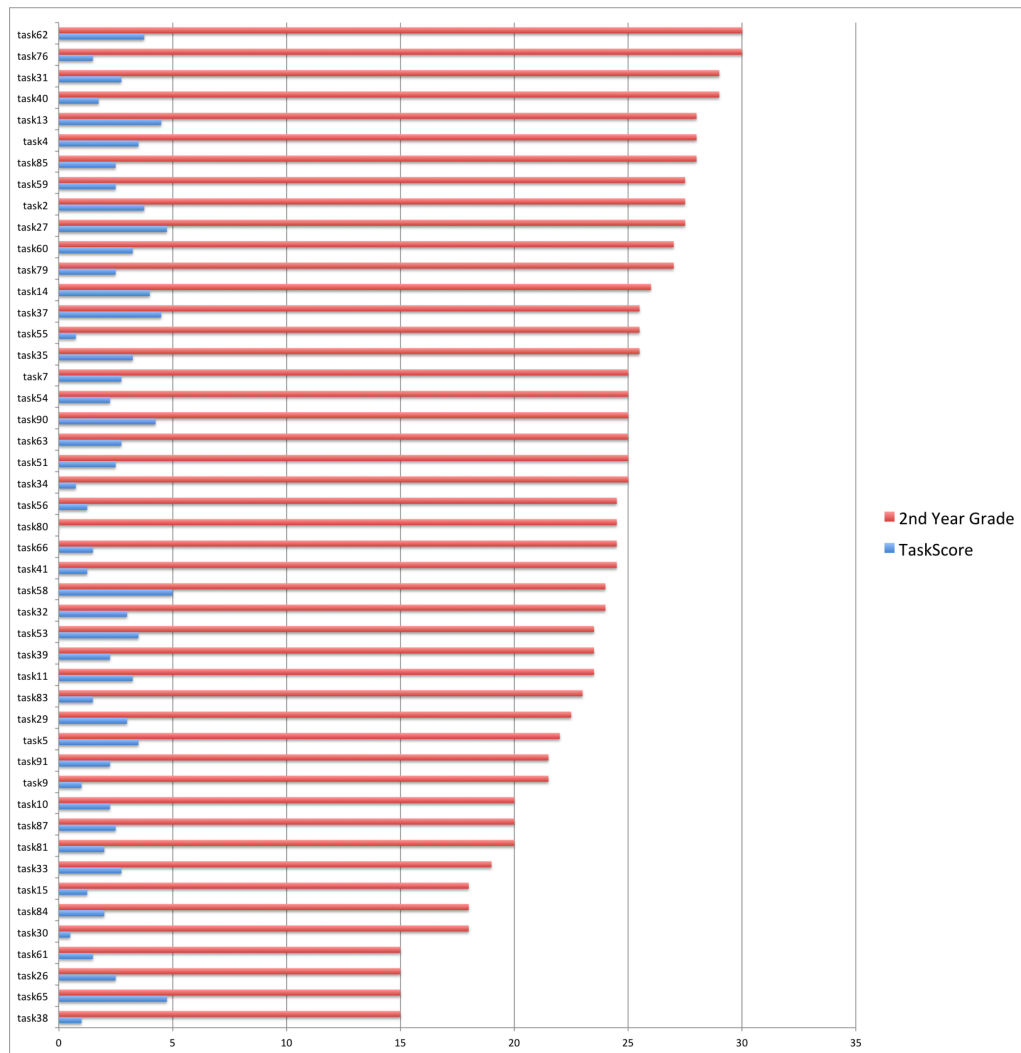


Figure 4.11: task score and second-year exam grade

The overall grade assigned to each student in the task was also compared to the grade they obtained (at the end of the semester) in the final exam of the course during which the experiment was conducted (cf. 3.1.4);⁵ this comparison was made in an attempt to determine whether students who were better translators were also the ones who had performed better in the proposed task (see Figure 4.11).⁶ Correlation analysis this time suggests a statistically significant ($p\text{-value} = 0.03$), weak positive correlation between the two scores (Kendall Tau

⁵ Great care was taken to respect the anonymity of students, the first-year exam grades were obtained directly by the lecturer of the second-year course.

⁶ Only 48 students (out of the 58 participating in the experiment) were included in this analysis, the remaining 10 had not taken the final exam yet when this work was completed.

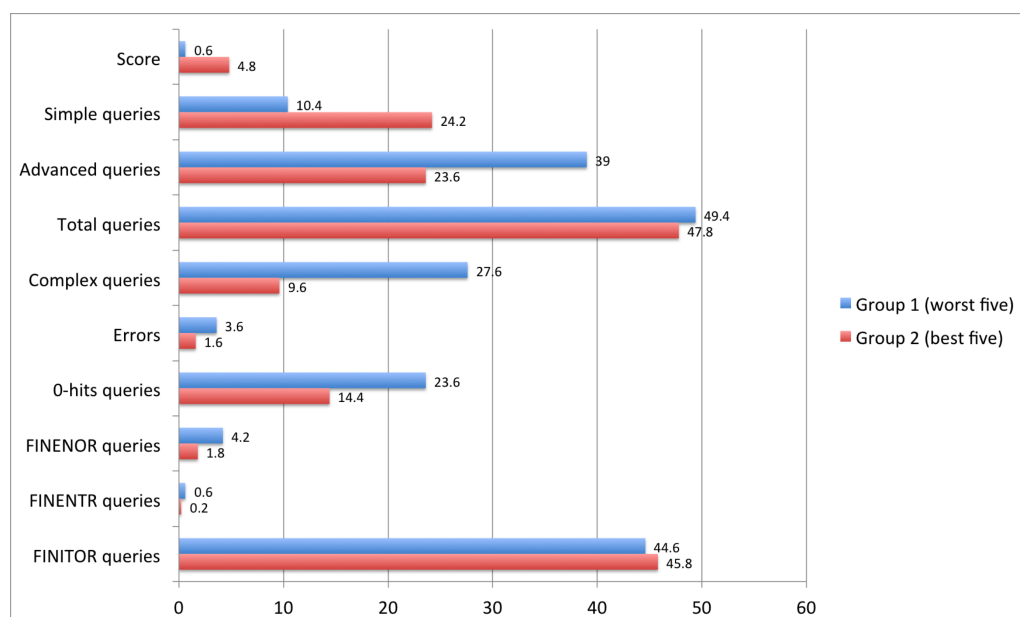


Figure 4.12: worst five vs. best five scoring students

= 0.21), which seems to indicate that students who obtain higher grades in translation are also more capable of using corpora to their advantage during a documentation task like the one we proposed (see below for a discussion on possible reasons for this).

4.3.3.2 Best-scoring students vs. worst-scoring students

A total of ten participants will be taken into consideration in this part of the analysis: the five students who obtained the higher grades and the five students who were assigned the lower grades. The query logs stored on the servers' databases made it possible to examine the users' behaviour rather closely, Figure 4.12 summarises the findings of this analysis. The bars in the charts represent the mean values for some of the aspects that were analysed: "score" is obviously the mean score obtained by the members of the group (0.6 for the worst students, 4.8 for the best ones). The data show no significant difference in the total number of queries, although the type of query does show a disparity:

group 1 used nearly twice as many advanced queries as group 2 ("advanced" queries are those written using the CQP syntax).⁷

Closer inspection of the actual queries submitted revealed that many of those that were typed in the "advanced" editors were actually rather basic ones, for example many of them consisted of simple word searches such as:

(f) `[word="utili"]`

(g) `[word="valore"] [word="netto"]`.

Although these fall under the definition of "queries that use the CQP syntax" they are not really advanced because they do the same thing that could have been done in the simple editor, they just employ a more convoluted syntax to do it. It was thus decided to isolate the queries that use the CQP syntax and that actually take advantage of its potential: if a query exploited the annotation of the corpus (part-of-speech and lemma) or used regular expressions over tokens, then it was classified as "complex". Examples of complex queries are:

(h) `[pos="NOUN"] [word="operativo"]`

(i) `[word="leasing"] [pos="ADJ"]`

Somewhat surprisingly, group 1 (worst scoring students) submitted an average of 27.6 complex queries (or 51.3% of all queries), while group 2 used an average of 9.6 complex queries (or 19.1% of all queries). Predictably enough, since it is easier to make mistakes using a more complex syntax, group 1 made more errors than group 2. Many of the recorded errors were caused by missing brackets and quotes, but a few were particularly interesting because they drew attention to a widespread phenomenon; for example this query

(j) `[noun="quando"] [] [word="strumento"]`.

is an error because there is no positional attribute "noun", what the user wanted to express was probably this:

⁷ For the purposes of this analysis, queries submitted using CQPWeb's "cqp" mode and Carcass's "Smart query" and "Expert query" are all considered "advanced".

```
(k) [word="quando" & pos="NOUN"] [] [word="strumento"]
```

Query (j) is one of many examples of queries written by students who were trying too hard to exploit the advanced features offered by the software that had been presented to them and were losing sight of what they were doing. Besides being syntactically wrong the query does not make much sense: "quando" (= "when" in English) is normally an adverb or a conjunction in Italian, it can be used as a noun only in very specific cases (e.g. "il come e il quando" = "how and when"). There seems to be no good reason for the student to be looking for "quando" as a noun followed by any word and then by the word "strumento" (= "instrument").

Also, the number of o-hits queries in group 1 is considerably higher than those of group 2 (47.7% vs. 30.1% of all queries, the mean of o-hits for the sample population is 35.8%). Manual inspection of the logs revealed that many of the complex queries were formally correct but substantially wrong, especially because of wrong part-of-speech tags. For example the following were syntactically correct:

```
(l) [pos="noun"] [lem="operativo"]
```

```
(m) [pos="preposition"] [word="operativi"]
```

but they returned 0 hits because in (l) the correct tag for "noun" is "NOUN" in uppercase and in (m) a preposition is represented by "PRE".

Figure 4.12 could suggest that there might be a correlation between scores obtained in the task by the students in the two groups and the number of complex queries performed: the higher their score in the test, the less likely they were to use complex queries. Correlation analysis suggests a weak negative correlation (Kendall's tau = -0.125) which however is not significant ($p > 0.5$). It is worth noting that the sample was very small, only 10 students, so it is possible that extending the analysis to the whole sample population (58 users) results could be more reliable and, possibly, statistically significant, but I will leave this for future work.

A number of insights were nonetheless gained from the manual inspection of query logs. Group 1 users invested a great deal of time refining queries that often returned 0 hits, while group 2 users spent more time devising different search strategies and reading concordance lines (server logs contain timestamps for all submitted queries, so it was possible to determine roughly how much time a user presumably spent perusing the results before trying a different strategy or moving on to the next expression).

The time constraint had of course a strong impact and students with more efficient time management skills were rewarded by better results: during the task, every five minutes participants were advised to move on to the next expression, whether that had found an answer or not. According to the server logs, students in group 2 heeded the advice more than those in group 1, they moved on to the next expression and, in many cases, found it rather quickly and used the extra time to go back and work some more on the previous one.

Distraction was another important factor: apart from the common mistake of using the CQP syntax in the simple editor, some users wasted time looking in the wrong corpus, user 80 for instance spent four minutes looking for Italian words in the English corpus. User 30 did not fully understand the briefing and was apparently convinced that FINITOR contained the translation of the text they had been given:⁸ in the context for the English expression "under operating leases" there was a date, September 30, 2010; analysing the query logs it is apparent that the student spent the first six minutes searching FINITOR for all possible combinations of "settembre", "2010" and "30".

Finally and perhaps most importantly users in group 2 employed a wider range of strategies, often trying new avenues (especially looking for translation of collocates) when results did not offer an answer.

⁸ Participants had been told that the examples had been taken from EasyJet's "Annual report and accounts 2010" and that the text was not in the corpora. They had also been given a hard copy of the pages where the expressions could be found (cf. 3.1.2).

Conversely, users in group 1 mostly kept refining the same queries over and over again, usually looking for evidence that supported intuitions that were often wrong instead of trying to infer answers from the context.

In conclusion the analysis of the behaviour of these two subsets of users showed that students who favoured a simpler approach were rewarded by a higher score. One possible explanation for this is that using simple queries reduced the number of errors and of o-hits queries, which in turn reduced the need for reformulation and refinement and left students more time to browse concordances.

In the light of these considerations, we might also try to explain why students who obtained a higher grade in the exam tended to perform better in the task (cf. 4.3.3.1): maybe they had a higher linguistic competence and they already knew the equivalents of some of the expressions (or of some parts of them), so all they needed to do was verify their intuitions; this could explain the lower number of complex queries. Or it could be the other way around: since the exam was taken after the experiment, their grades were better at the end of the semester precisely because they had developed more efficient strategies of finding terminology and correct turns of phrases using corpora; strategies that possibly only include a minimal use of complex queries using grammatical patterns (although as we said above, further work is needed to determine if there is indeed a correlation between the task score and the number of complex queries).

CONCLUSIONS

General conclusions

In this thesis I started by presenting an overview of the available software solutions for corpus analysis and proceeded to explain why they appear inadequate to satisfy the needs of corpus users.

I then introduced a new open-source system designed to overcome the limitations of current corpus tools and explained the advantages this new architecture has to offer to users and corpus builders alike.

After describing the new architecture, I proposed an experiment aimed at evaluating the usability of the new system by comparing it to a well-known corpus tool, but also at observing the way in which a group of users (more specifically post-graduate translation students) interacted with corpora via these two corpus analysis tools.

Appraisal of the work

My goal for this thesis was surveying the state of the art in the field of corpus query tools and then designing and developing the foundations of a new system that could overcome the limitations of current software solutions.

The goal has been achieved at least partially: the most crucial drawbacks of available tools have been identified, the groundwork for a new system is in place and a first usable version of the software has

been successfully tested. Whether the new architecture will actually be able to overcome the limitations of existing corpus tools remains to be seen: I am convinced that the tool has the potential to become something really unique, but its future depends largely on the time and resources that will be devoted to its development, as well as on the interest that it will be able to generate in the community.

Impact and beneficiaries of the work

The work presented in this thesis can be beneficial from three different points of view: first of all, we offer a perspective on how corpus users interact with corpora; the analysis of the query logs collected during the experiment and presented in chapter 4 revealed a series of shortcomings in the preparation of participants: the nature of these shortcomings seems to suggest that a greater effort should probably be put into teaching translators how to use corpora to effectively find terminology and correct turns of phrases.

Another interesting perspective is represented by the results of questionnaires 1, 2 and 3 which revealed quite clearly some of the preferences and dislikes in terms of user interfaces expressed by corpus users: these data could undoubtedly be valuable to software developers to improve existing software tools.

Finally, thanks to this work, people interested in consulting corpora will be able to use the new tool that was developed and made available to the community: even though the new application is not complete and still a bit rough around the edges, it is quite usable and it is in fact already being used internally by students and faculty of the University of Bologna. For instance, Sarcophagus was recently used to make available a corpus of press releases created by students: they had originally used AntConc, but as new texts were added during the course of the term, it became increasingly difficult to use the corpus effectively with AntConc. Deployment of the corpus on Corpse was rather easy, and

in this way the resource became instantly available to all students in the class.

Limitations of the work and possible extensions

The choice of a group of translators as experimental subject was dictated by necessity: the only large group of people I had access to were translators. Although there is nothing wrong with this group of users as they are a perfect examples of the kind of non-specialist users that could benefit from a greater access to corpora, it would have been interesting to have a more varied population. The type of task that was assigned was also largely the result of a compromise: everything had to be completed in the space of two lessons so there was no time to assign a real translation for instance.

Moreover, in an ideal world, the comparison would have been made using more than just one alternative tool (CQPWeb): the Sketch Engine and the BYU corpora present some incredibly advanced features that it would have been very interesting to test. Maybe I could have obtained permission to upload the three corpora required for the experiment at least to the Sketch Engine, but at the time the endeavour appeared daunting because of the many technical problems of setting up such an experiment, which required, among other things, some way of obtaining the logs of all queries from the system administrator.

Plans for future developments of the system

As far as the future of the development of Sarcophagus is concerned, the experiment clearly showed that there is still room for improvement in the query editors, especially to avoid the subtle mistakes which do not result in syntax errors but still produce o-hits queries: integrating a full CQP parser into the Expert Query Editor would undoubtedly be extremely beneficial in this respect. User comments also gave a clear

indication that the result visualisation panel needs to be rewritten in order to become less confusing.

In addition to these improvements, quite a few features need to be implemented before the system can be made publicly available, including the support for parallel corpora, frequency lists and keywords extraction as well as the ability to find collocations. Luckily, the type of approach taken with the development of this system opens many avenues of integration with existing tools and libraries, such as for instance NLTK (a set of tools for natural language processing), so not all the work will have to be done from scratch. When the possibility to upload corpora to the system is added, Carcass will also be integrated with BootCaT, a program for the semi-automatic creation of web corpora.

Concluding remarks

Software tools for analysing corpora have been available for a few decades now and although technology has changed dramatically and the performance and features of current tools would have been unimaginable when Roberto Busa started working on his *Index Thomisticus*, some of the fundamental problems of corpus linguistics still appear unresolved. The situation has improved in many respects but sharing data remains problematic (albeit now the main complications are of a legal and not technical nature), tools continue to be developed ad hoc for specific corpora and at least a part of the people who could benefit from the use of corpora are still unable to consult them because of technical difficulties.

I hope that the insights and suggestions contained in this work and the new software I developed will contribute to a greater usability and more widespread availability of corpora.

REFERENCES

- Abel, A., S. Anstein and S. Petrakis (2009): "Die Initiative Korpus Südtirol", in *Linguistik online* 38.2.
- Anstein, S. (2009): "Comparing Linguistic Varieties on the Basis of Corpora - the Vis-À-Vis Architecture", in *GSCL Workshop: Linguistic Processing Pipelines*, Potsdam, Germany.
- Anthony, L. (2005): "AntConc: A Learner and Classroom Friendly, Multi-Platform Corpus Analysis Toolkit", in *Proceedings of IWLeL 2004: An Interactive Workshop on Language e-Learning*, Tokyo.
- Ari, O. (2006): "Review of three software programs designed to identify lexical bundles", in *Language Learning & Technology* 10.1, 30-37.
- Aston G. & L. Burnard (1998): "The BNC handbook: exploring the British National Corpus with SARA", Edinburgh: Edinburgh University Press.
- Aston, G. & D. Rodi (2012): "Speech corpora for language learning", in *Proceedings of the 10th Teaching and Language Corpora Conference*, Warsaw.
- Attardi, G. (2005): "IXE at the TREC Terabyte Task", in *Proceedings of The Forteenth Text Retrieval Conference*, Gaithersburg.
- Barbera, M., E. Corino & C. Onesti (2007): "Corpora e linguistica in rete", Perugia: Guerra Edizioni.
- Barlow, M. (2002): "ParaConc: Concordance software for multilingual parallelcorpora", in *Language Resources for Translation Work and Research, LREC 2002*, Las Palmas de Gran Canaria.

- Baroni, M. & S. Bernardini (2004): "BootCaT: Bootstrapping corpora and terms from the web", in *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon.
- Baroni, M., S. Bernardini, A. Ferraresi & E. Zanchetta (2009): "The WaCky Wide Web: A Collection of Very Large Linguistically Processed Web-Crawled Corpora", in *Language Resources and Evaluation* .43, 209-226.
- Baroni, M., S. Bernardini, F. Comastri, L. Piccioni, A. Volpi, G. Aston & M. Mazzoleni (2004): "Introducing the "la Repubblica" corpus: A large, annotated, TEI(XML)-compliant corpus of newspaper Italian", in *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon.
- Bernardini, S. & S. Castagnoli (2008): "Corpora for translator education and translation practice", in Yuste Rodrigo, E. (ed.), *Topics in Language Resources for Translation and Localisation*, Amsterdam & Philadelphia: John Benjamins.
- Bernardini, S., S. Castagnoli, A. Ferraresi, F. Gaspari & E. Zanchetta (2011): "Turning Wikipedia into Comparapedia: Towards a new type of comparable corpus for language professionals", in *Corpus Linguistics 2011*, University of Birmingham.
- Boleda, G., S. Bott, B. Poblete, C. Castillo, M.E.Fuenmayor, T. Badia & V. López (2004): "CucWeb: un corpus del català construït a partir de la web", in *II Congrés Online de l'Observatori per a la Cibersocietat*, Barcelona.
- Borghetti, C., M. Brunello & S. Castagnoli (forthcoming): "I generi del web tra tradizione e innovazione: un'analisi linguistica sulla base del corpus PAISÀ", in Cerruti, M. (ed.), *Scritto e parlato, formale e informale: La comunicazione mediata dalla rete*, Roma: Carocci.
- Boulton A. (2012): "Data-driven learning: on paper, in practice", in Harris T. & M. Moreno Jaén (ed.), *Corpus Linguistics in Language Teaching*, Bern: Peter Lang.
- Brooke, J. (1996): "SUS: A 'quick and dirty' usability scale", in P. W. Jordan, B. Thomas, B. A. Weerdmeester, & I. L. McClelland (ed.), *Usability evaluation in industry*, London: Taylor & Francis.
- Burnard, L. (2007): "Reference Guide for the British National Corpus (XML Edition)", available at: <http://www.natcorp.ox.ac.uk/>.

- Christ, O. (1994): "A modular and flexible architecture for an integrated corpus query system", in *Papers in Computational Lexicography (COMPLEX '94)*, Budapest.
- Davies, M. (2008): "The Corpus of Contemporary American English: 425 million words, 1990-present", *available at*: <http://corpus.byu.edu/coca/>.
- Evert, S. (2008): "Inside the IMS Corpus Workbench", *available at*: http://cwb.sourceforge.net/files/Evert2008_InsideCWB.pdf.
- Fletcher, W. H. (2007): "Implementing a BNC-compare-able web corpus", in Fairon, C., H. Naets, A. Kilgarrieff & G.-M. De Schryver (ed.), *Building and exploring web corpora*, Louvain-la-Neuve: Cahiers du Cental.
- Fletcher, W.H. (2004): "Facilitating Compilation and Dissemination of Ad Hoc Web Corpora", in Aston, G, S. Bernardini & D. Stewart (ed.), *Papers from the Fifth International Conference on Teaching and Language Corpora*, Amsterdam: Benjamins.
- Frederick, S., C. Ramsay, S. Blades & N. White (2010): "Learning Ext JS 3.2", Birmingham: Packt.
- Gaspari F. & S. Bernardini (2010): "Comparing Non-native and Translated Language: Monolingual Comparable Corpora with a Twist", in Xiao, R. (ed.), , Newcastle: Cambridge Scholars Publishing.
- Hafner, C. & C. Candlin (2007): "Corpus tools as an affordance to learning in professional legal education", in *Journal of English for Academic Purposes* 6.4, 303-318.
- Hardie, A. (forthcoming): "CQPweb – combining power, flexibility and usability in a corpus analysis tool".
- Hoffmann, S. & S. Evert (2006): "BNCweb (CQP-edition): The marriage of two corpus tools", in Braun, S., K. Kohn & J. Mukherjee (ed.), *Corpus Technology and Language Pedagogy: New Resources, New Tools, New Methods*, Frankfurt am Main: Peter Lang.
- Hoffmann, S., Evert, S., Smith, N., Lee, D. & B. Prytz (2008): "Corpus Linguistics with BNCweb: A Practical Guide", Frankfurt am Main: Peter Lang.
- Kereki, F. (2010): "Essential GWT: Building for the Web with Google Web Toolkit 2", Boston: Addison Wesley.

- Kilgariff A. & G. Grefenstette (2003): "Introduction to the Special Issue on the Web as Corpus", in *Computational Linguistics - Special issue on web as corpus* 29.3, 333-347.
- Kilgariff, A., P. Rychly, P. Smrz & D. Tugwell (2004): "The Sketch Engine", in *Proceedings of EURALEX 2004*, Lorient.
- Lee, D. Y. W. and P. Rayson (2000): "Xkwic: a powerful concordancer for research", in *In Workshop at Teaching and Language Corpora conference (TALC2000)*, Graz, Austria.
- Lehmann, H.-M., P. Schneider & S. Hoffmann (2000): "BNCweb", in (ed.), *Corpora Galore: Analysis and Techniques in Describing English*, Amsterdam: Rodopi.
- Lidwell W., K. Holden & J. Butler (2003): "Universal Principles of Design", Beverly US-MA: Rockport Publishers.
- Marcus M., B. Santorini & M.A. Marcinkiewicz (1993): "Building a large annotated corpus of English: The Penn Treebank", in *Computational Linguistics - Special issue on using large corpora: II* 19.2, 313-330.
- McCandless, M., E. Hatcher & Otis Gospodnetic (2010): "Lucene in Action, Second Edition", Stamford: Manning Publications.
- McEnery T. & A. Hardie (2012): "Corpus Linguistics: Method, Theory and Practice", Cambridge: Cambridge University Press.
- Nielsen, J. (1994): "Heuristic evaluation", in Nielsen, J. & Mack, R.L. (ed.), *Usability Inspection Methods*, New York: John Wiley & Sons.
- Onelli, C., D. Proietti, C. Seidenari & F. Tamburini (2006): "The DiaCORIS Project: a diachronic corpus of written Italian", in *Proceedings of the 5th International Conference on Language Resources and Evaluation - LREC 2006*, Genoa.
- Petri, J. (2010): "NetBeans Platform 6.9 Developer's Guide", Birmingham: Packt Publishing Ltd.
- Piccioni S. (2008): "Convenzionalità e creatività della metafora: il caso di "NOME de NOME" in Federico García Lorca", Ph.D. Thesis. Università di Bologna: Italy.
- Rossini Favretti R., F. Tamburini & E. Martelli (2001): "Words from Bononia Legal Corpus", in *International Journal of Corpus Linguistics* 6.Special Issue, 13-34

- Rossini Favretti R., F. Tamburini, C. De Sanctis (2001): "A corpus of written Italian: a defined and dynamic model", in *Proceedings of Corpus Linguistics 2001*, Lancaster, UK.
- Russo, M., C. Bendazzoli & A. Sandrelli (2006): "Looking for lexical patterns in a trilingual corpus of source and interpreted speeches: extended analysis of EPIC (European Parliament Interpreting Corpus)", in (ed.), *FORUM, International journal of interpretation and translation IV (1)*.
- Rychlý, P. (2007): "Manatee/Bonito - A Modular Corpus Manager", in *1st Workshop on Recent Advances in Slavonic Natural Language Processing*, Brno.
- Santos, D. & A. Frankenberg-Garcia (2007): "The corpus, its users and their needs: A user-oriented evaluation of COMPARA", in *International Journal of Corpus Linguistics* 12.3, 335-374.
- Schmid, H. (1994): "Probabilistic Part-of-Speech Tagging Using Decision Trees", in *Proceedings of International Conference on New Methods in Language Processing*, Manchester.
- Scott, M. (2008): "WordSmith Tools version 5", in (ed.), , Liverpool: Lexical Analysis Software.
- Sharoff S. (2006): "Creating general-purpose corpora using automated search engine queries", in Baroni, M. & S. Bernardini (ed.), *Wacky! Working papers on the Web as Corpus*, Bologna: Gedit.
- Simpson, R., S.L. Briggs, J. Ovens, & J.M. Swales (2002): "The Michigan corpus of academic spoken English", in (ed.), , Ann Arbor, Michigan: The Regents of the University of Michigan.
- Smith, N., S. Hoffmann & P. Rayson (2008): "Corpus Tools and Methods, Today and Tomorrow: Incorporating Linguists' Manual Annotations", in *Literary and Linguistic Computing* 23.2, 163-180.
- Wiechmann, D. & S. Fuhs (2006): "Concordancing software", in *Corpus Linguistics and Linguistics Theory* 2.1, 107-127.

APPENDIX A

A printout of this set of instruction was given to each participant in the experiment. Instructions were slightly different for each of the four groups, in particular the order of the systems to be used changed, as did the order of the expressions to research. Each sheet had been further customised with a unique username which was also used to match the task results (see Appendix E) with the three questionnaires and the query logs

Istruzioni (gruppo X)

Introduzione – Il vostro compito

Nei brani che vi abbiamo consegnato sono state evidenziate delle espressioni. Le espressioni e le frasi che le contengono sono riportate anche nella "Tabella Risultati".

Per ciascuna delle espressioni inglesi fornite, utilizzando **esclusivamente** lo strumento informatico che vi sarà indicato di volta in volta:

- cercate di capire il significato dell'espressione
- trovate l'espressione italiana equivalente
- riportate l'espressione italiana nella "Tabella Risultati"

- cercate un esempio di uso rilevante dell'espressione italiana, idealmente vorremo una frase breve, simile alla frase breve inglese che vi abbiamo fornito
- spiegate brevemente come siete arrivati a trovare l'equivalente italiano
- spiegate brevemente in base a quali criteri avete scelto l'esempio italiano

Non è permesso utilizzare dizionari e/o glossari o accedere ad altre pagine Web.

Credenziali d'accesso a CQPWeb:

username:

password:

Parte 1 – Warm-up (10 minuti)

- Usando **Carcass** individuate l'espressione italiana equivalente a:
 - functional currency (pag. 61)
- riportate nella "Tabella Risultati" l'espressione italiana
- usando **CQPWeb** trovate un esempio rilevante (ossia una frase breve, anche incompleta) di uso dell'espressione italiana che avete individuato e riportatela nella "Tabella Risultati"
- spiegate brevemente come siete arrivati a trovare l'equivalente italiano
- spiegate brevemente in base a quali criteri avete scelto l'esempio italiano

Parte 2 (15 minuti)

- Usando **CQPWeb** individuate le espressioni italiane equivalenti a:

- assets given (pag. 61)
- diluted earnings per share (pag. 70)
- intangible assets (pag. 61)
- trovate un esempio rilevante (ossia una frase breve, anche incompleta) di uso di ciascuna delle espressioni italiane che avete individuato e incollatele nella "Tabella Risultati"
- spiegate brevemente come siete arrivati a trovare l'equivalente italiano
- spiegate brevemente in base a quali criteri avete scelto l'esempio italiano

Questionario 1 (5 minuti)

- aprite la pagina web <http://goo.gl/fQTfO> e compilate il "Questionario 1"

Parte 3 (15 minuti)

- Usando **Carcass** individuate le espressioni italiane equivalenti a:
 - under operating leases (pag. 73)
 - net book value (pag. 73)
 - hedging instrument (pag. 63)
- trovate un esempio rilevante (ossia una frase breve, anche incompleta) di uso di ciascuna delle espressioni italiane che avete individuato e incollatele nella "Tabella Risultati"
- spiegate brevemente come siete arrivati a trovare l'equivalente italiano
- spiegate brevemente in base a quali criteri avete scelto l'esempio italiano

Questionario 2 (5 minuti)

- aprite la pagina web <http://goo.gl/JU4vO> e compilate il "Questionario 2"

Questionario 3 (10 minuti)

- aprite la pagina web <http://goo.gl/8JTcF> e compilate il "Questionario 3"

APPENDIX B

This is the 4-page extract of the the "EasyJet Annual report and accounts 2010" that was given to the participants, it includes pages 61, 63, 70 and 73. The complete text is available online at <http://goo.gl/eAoWI>.

Basis of consolidation

The consolidated accounts incorporate those of easyJet plc and its subsidiaries for the years ended 30 September 2009 and 2010.

A subsidiary is an entity controlled by easyJet. Control exists when easyJet has the power, directly or indirectly, to govern the financial and operating policies of an entity so as to benefit from its activities.

Intragroup balances, transactions and any unrealised gains and losses arising from intragroup transactions are eliminated in preparing the consolidated accounts.

Foreign currencies

The primary economic environment in which a subsidiary operates determines its **functional currency**. The consolidated accounts of easyJet are presented in sterling, which is the Company's functional currency and the Group's presentation currency. Certain subsidiaries have operations that are primarily influenced by a currency other than sterling. Exchange differences arising on the translation of these foreign operations are taken to reserves until all or part of the interest is sold, when the relevant portion of the exchange gains or losses is recognised in the income statement. Profits and losses of foreign operations are translated into sterling at average rates of exchange during the year, since this approximates the rates on the dates of the transactions.

Transactions arising in foreign currencies are recorded using the rate of exchange ruling at the date of the transaction. Monetary assets and liabilities denominated in foreign currencies are translated into sterling using the rate of exchange ruling at the balance sheet date and (except where the asset or liability is designated as a cash flow hedge) the gains or losses on translation are included in the income statement. Non-monetary assets and liabilities denominated in foreign currencies are translated into sterling at foreign exchange rates ruling at the dates the transactions were effected.

Revenue recognition

Revenues comprise the invoiced value of airline services (net of air passenger duty, VAT and discounts), and ancillary revenue.

Passenger revenue arises from the sale of flight seats and is recognised when the service is provided. Unearned revenue represents flight seats sold but not yet flown and is included in trade and other payables until it is realised in the income statement when the service is provided.

Ancillary revenue is generally recognised when the flight to which it relates departs. Certain types of ancillary revenue are recognised at the time the benefit of the service provided passes to the customer.

Amounts paid by "no-show" customers are recognised as passenger or ancillary revenue as appropriate when the booked service is provided as such customers are not generally entitled to change flights or seek refunds once a flight has departed.

Business combinations

Business combinations in prior years were accounted for by applying the purchase method. The cost of the acquisition was measured at the aggregate of the fair values, at the date of exchange, of **assets given** and liabilities incurred or assumed plus any costs directly attributable to the business combination. The acquiree's identifiable assets and liabilities were recognised at their fair values at the acquisition date. There have been no business combinations since the effective date of IFRS 3 Business Combinations (Revised).

Goodwill arising on acquisition is recognised as an asset and initially measured at cost, being the excess of the cost of the business combination over easyJet's interest in the net fair value of the identifiable assets, liabilities and contingent liabilities recognised.

Goodwill and other intangible assets

Goodwill is stated at cost less any accumulated impairment losses. It has an indefinite expected useful life and is tested for impairment at least annually or where there is any indication of impairment.

Landing rights are stated at cost less any accumulated impairment losses. They are considered to have an indefinite useful life as they will remain available for use for the foreseeable future provided minimum utilisation requirements are observed, and are tested for impairment at least annually or where there is any indication of impairment.

Other **intangible assets** are stated at cost less accumulated amortisation, which is calculated to write-off their cost, less estimated residual value, on a straight-line basis over their expected useful lives. Expected useful lives and residual values are reviewed annually.

	Expected useful life
Computer software	3 years
Contractual rights	Over the length of the related contracts

Financial instruments

Financial instruments are recognised when easyJet becomes a party to the contractual provisions of the relevant instrument and derecognised when it ceases to be a party to such provisions.

Where market values are not available, the fair value of financial instruments is calculated by discounting cash flows at prevailing interest rates and by applying year end exchange rates.

Non-derivative financial assets

Non-derivative financial assets are recorded at amortised cost and include loan notes, trade receivables, cash and money market deposits. Investments in equity instruments are carried at cost where fair value cannot be reliably measured due to significant variability in the range of reasonable fair value estimates.

Restricted cash comprises cash deposits which have restrictions governing their use and is classified as a current or non-current asset based on the estimated remaining length of the restriction. Cash and cash equivalents comprise cash held in bank accounts with no access restrictions and bank or money market deposits repayable on demand or maturing within three months of inception. Interest income on cash and money market deposits is recognised using the effective interest method.

Impairment losses are recognised on financial assets carried at amortised cost where there is objective evidence that an impairment loss has been incurred. The amount of the loss is measured as the difference between the asset's carrying amount and the present value of future cash flows, discounted at the original effective interest rate.

If, subsequently, the amount of the impairment loss decreases, and the decrease can be related objectively to an event that occurred after the impairment was recognised, the appropriate portion of the loss is reversed. Both impairment losses and reversals are recognised in the income statement as components of net finance charges.

Non-derivative financial liabilities

Non-derivative financial liabilities are initially recorded at fair value less directly attributable transaction costs, and subsequently at amortised cost. Interest expense on loans is recognised using the effective interest method.

Borrowings are classified as current liabilities unless there is an unconditional right to defer settlement of the liability for at least 12 months after the balance sheet date.

Derivative financial instruments

Derivative financial instruments are measured at fair value.

Derivative financial instruments designated as cash flow hedges are used to mitigate operating and investing transaction exposures to movements in jet fuel prices and currency exchange rates. Hedge accounting is applied to these instruments.

Changes in intrinsic fair value are recognised in other comprehensive income to the extent that the cash flow hedges are determined to be effective. All other changes in fair value are recognised immediately in the income statement. Where the hedged item results in a non-financial asset or liability the accumulated gains and losses previously recognised in other comprehensive income form part of the initial carrying amount of the asset or liability. Otherwise accumulated gains and losses are recognised in the income statement in the same period in which the hedged items affect the income statement.

Hedge accounting is discontinued when a **hedging instrument** is derecognised (e.g. through expiry or disposal), or no longer qualifies for hedge accounting. Where the hedged item is a highly probable forecast transaction, the related gains and losses remain in other comprehensive income until the transaction takes place.

When a hedged future transaction is no longer expected to occur, any related gains and losses previously recognised in other comprehensive income are immediately recognised in the income statement.

Financial guarantees

If a claim on a financial guarantee given to a third party becomes probable, the obligation is recognised at fair value. For subsequent measurement, the carrying amount is the higher of initial measurement and best estimate of the expenditure required to settle the obligation on the statement of financial position date.

Notes to the accounts

continued

5 Tax charge / (credit) continued

Deferred tax

The net deferred tax liability included in the balance sheet is as follows;

	Accelerated capital allowances £ million	Short-term timing differences £ million	Tax losses £ million	Fair value (gains)/losses £ million	Share-based payments £ million	Total £ million
At 1 October 2009	35.5	51.7	(16.0)	11.2	(6.1)	76.3
Charged / (credited) to the income statement	26.2	5.6	15.8	(0.4)	(0.8)	46.4
Charged to other comprehensive income	–	–	–	22.5	–	22.5
Charged to shareholders' equity	–	–	–	–	2.7	2.7
At 30 September 2010	61.7	57.3	(0.2)	33.3	(4.2)	147.9

	Accelerated capital allowances £ million	Short-term timing differences £ million	Tax losses £ million	Fair value (gains)/losses £ million	Share-based payments £ million	Total £ million
At 1 October 2008	49.7	30.3	–	31.1	(3.8)	107.3
Charged / (credited) to the income statement	(14.2)	23.3	(16.0)	–	(1.2)	(8.1)
Transfer from current tax liabilities	–	(1.9)	–	–	–	(1.9)
Credited to other comprehensive income	–	–	–	(19.9)	–	(19.9)
Credited to shareholders' equity	–	–	–	–	(1.1)	(1.1)
At 30 September 2009	35.5	51.7	(16.0)	11.2	(6.1)	76.3

It is estimated that deferred tax liabilities of approximately £12.2 million (2009: assets of £7.4 million) will reverse during the next financial year. Deferred tax assets and liabilities have been offset where they relate to taxes levied by the same taxation authority. There are no unrecognised deferred tax assets.

6 Earnings per share

Basic earnings per share has been calculated by dividing the profit for the year by the weighted average number of shares in issue during the year after adjusting for shares held in employee share trusts.

For **diluted earnings per share**, the weighted average number of ordinary shares in issue is adjusted to assume conversion of all dilutive potential shares. Share options granted to employees where the exercise price is less than the average market price of the Company's ordinary shares during the year are considered to be dilutive potential shares. Where share options are exercisable based on performance criteria and those performance criteria have been met during the year, these options are included in the calculation of dilutive potential shares.

Earnings per share is based on:

	2010 £ million	2009 £ million
Profit for the year	121.3	71.2
	2010 million	2009 million
Weighted average number of ordinary shares in issue during the year used to calculate basic earnings per share	426.5	421.9
Weighted average number of dilutive share options	6.0	6.4
Weighted average number of ordinary shares used to calculate diluted earnings per share	432.5	428.3
Earnings per share	2010 pence	2009 pence
Basic	28.4	16.9
Diluted	28.0	16.6

During the year ended 30 September 2010, six aircraft were sold and leased back **under operating leases**. Two of these aircraft were acquired during the year ended 30 September 2009. The amounts shown above under the caption "aircraft sold and leased back" relate to these two aircraft and deposits paid on the other four aircraft before 1 October 2009.

During the year ended 30 September 2009, five A319 aircraft were transferred back to property, plant and equipment from assets held for sale.

The **net book value** of aircraft includes £153.2 million (2009: £148.5 million) relating to advance and option payments for future deliveries of aircraft. This amount is not depreciated.

Aircraft with a net book value of £1,107.6 million (2009: £984.5 million) are mortgaged to lenders as loan security.

Aircraft with a net book value of £105.4 million (2009: £71.1 million) are held under finance leases.

easyJet is contractually committed to the acquisition of 47 (2009: 74) Airbus A320 family aircraft with a total list price of US\$2.2 billion (2009: US\$3.4 billion) before escalations and discounts, for delivery in the period to May 2013.

9 Loan notes

In 2001, easyJet in consortium with six other UK airlines formed The Airline Group Limited in order to acquire a non-controlling interest in NATS, the company that owns the UK air traffic control system. easyJet's investment is principally in the form of unsecured loan notes bearing interest at a fixed rate of 8%. Interest receivable is settled by the issue of additional loan notes. Redemption is governed by a priority agreement among the consortium members.

	2010 £ million	2009 £ million
At 1 October	12.6	12.0
Interest receivable converted to loan notes	1.1	0.9
Redemption of loan notes	(0.6)	(0.3)
At 30 September	13.1	12.6

10 Other non-current assets

	2010 £ million	2009 £ million
Recoverable supplemental rent on leased aircraft (pledged as collateral)	48.6	57.3
Deposits held by aircraft lessors	1.1	2.3
Other	3.8	3.1
	53.5	62.7

Supplemental rent is pledged to lessors to provide collateral should an aircraft be returned in a condition that does not meet the requirements of the lease and is refunded when qualifying heavy maintenance is performed, or is offset against the costs incurred at the end of the lease.

11 Assets held for sale

Following the acquisition of GB Airways in the year ended 30 September 2008, seven A321 aircraft were classified as assets held for sale.

During the year ended 30 September 2009 three of these aircraft were sold realising a net profit of £11.0 million. At 30 September 2009, easyJet continued to market the remaining four A321 aircraft and although the period over which the assets were classified as held for sale exceeded one year, the Directors considered that this classification remained appropriate.

easyJet has entered into an arrangement to dispose of the remaining four aircraft. Subsequent to the year end, in November 2010, the legal title to these four aircraft was transferred. The total cash consideration to be received is £75.2 million. easyJet has incurred certain costs in connection with the disposal and the aggregate net loss on the disposal of £7.0 million has been charged to the income statement in the year ended 30 September 2010.

APPENDIX C

A printout of this "cheat sheet" was given to all participants in the experiment. Its contents were fitted on a single page to provide a quick, easily manageable reference during the task.

Cheat sheet

Lem vs. Lemma

- Carcass utilizza lem
- CQPWeb utilizza lemma

Modalità di ricerca utilizzabili

- Carcass: Simple query, Expert query, Smart query
- CQPWeb: Simple query (ignore case), Simple query (case-sensitive), CQP syntax

Sintassi CQP (da utilizzare solo in modalità "Expert query" o "CQP syntax")

1. [word="investimento" %cd]
trova la parola "investimento" ignorando maiuscole/minuscole
2. [word="investimento" %cd][word="immobiliare" %cd]
trova la parola "investimento" seguita da "immobiliare" ignorando maiuscole/minuscole in entrambe le parole

3. [word="acquisi.*" %cd][pos="NOUN"]
trova parole che iniziano per "acquisi" seguite da un nome, ignorando maiuscole/minuscole
4. [lem="applicare"][pos="DET.*"][pos="NOUN"]
trova il verbo "applicare" (in tutte le sue forme) seguito da un determinante e da un nome
5. [lem="rigettare"][][pos="NOUN"]
trova il verbo "rigettare" (in tutte le sue forme) seguito da una parola qualsiasi e da un nome
6. [lem="rigettare"][][pos="NOUN"]
trova il verbo "rigettare" (in tutte le sue forme) seguito da zero o più parole qualsiasi fino a trovare il primo nome
7. [lem="rigettare"][][pos="NOUN"] within s
come l'esempio (6) ma limita la ricerca ad una frase ("s" sta per "sentence"), ossia non è possibile che "rigettare" sia in una frase e che il nome si trovi nella frase seguente
8. [lem="rigettare"][][pos="NOUN"] within 5
come l'esempio (6) ma il nome deve essere a non più di 5 parole da "rigettare"
9. [word="ricavato" & pos="NOUN" %cd]
trova la parola "ricavato" quando è un nome, ignora la differenza maiuscole/minuscole
10. [word="ricavato" & pos="VER.*" %cd]
trova la parola "ricavato" quando è un verbo, ignora la differenza maiuscole/minuscole

In caso di errore verificare che:

- sia stata scelta la modalità di query appropriata (i.e. "Expert query" su Carcass o "CQP syntax" su CQPWeb)
- parentesi quadre e virgolette siano state aperte/chiusure correttamente
- il pos sia stato scritto esattamente come riportato nel tagset (oppure usare .*)
- non sia stato usato "lem" in luogo di "lemma" o viceversa

APPENDIX D

The following two tables contain the tagsets of the three corpora made available to participants in the experiment. The two corpora in English (Finenor and Finentr) uses the same tagset while Finitor uses a different one.

Tagsets

FINENOR e FINENTR

CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential there
FW	Foreign word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker
MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PRP	Personal pronoun
PRP\$	Possessive pronoun (prolog version PRP-S)
RB	Adverb
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
SYM	Symbol

TO	to
UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non-3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	Wh-determiner
WP	Wh-pronoun
WP\$	Possessive wh-pronoun (prolog version WP-S)
WRB	Wh-adverb

FINITOR

ADJ	adjective
ADV	adverb (excluding -mente forms)
ADV:mente	adverb ending in -mente
ART	article
ARTPRE	preposition + article
AUX:fin	finite form of auxiliary
AUX:fin:cli	finite form of auxiliary with clitic
AUX:geru	gerundive form of auxiliary
AUX:geru:cli	gerundive form of auxiliary with clitic
AUX:infi	infinitival form of auxiliary
AUX:infi:cli	infinitival form of auxiliary with clitic
AUX:ppast	past participle of auxiliary
AUX:ppre	present participle of auxiliary
CHE	che
CLI	clitic
CON	conjunction
DET:demo	demonstrative determiner
DET:indef	indefinite determiner
DET:num	numeral determiner
DET:poss	possessive determiner
DET:wh	wh determiner
NEG	negation
NOCAT	non-linguistic element
NOUN	noun
NPR	proper noun
NUM	number
PRE	preposition
PRO:demo	demonstrative pronoun
PRO:indef	indefinite pronoun
PRO:num	numeral pronoun
PRO:pers	personal pronoun
PRO:poss	possessive pronoun
PUN	non-sentence-final punctuation mark
SENT	sentence-final punctuation mark
VER2:fin	finite form of modal/causal verb
VER2:fin:cli	finite form of modal/causal verb with clitic
VER2:geru	gerundive form of modal/causal verb

VER2:geru:cli	gerundive form of modal/causal verb with clitic
VER2:infi	infinitival form of modal/causal verb
VER2:infi:cli	infinitival form of modal/causal verb with clitic
VER2:ppast	past participle of modal/causal verb
VER2:ppre	present participle of modal/causal verb
VER:fin	finite form of verb
VER:fin:cli	finite form of verb with clitic
VER:geru	gerundive form of verb
VER:geru:cli	gerundive form of verb with clitic
VER:infi	infinitival form of verb
VER:infi:cli	infinitival form of verb with clitic
VER:ppast	past participle of verb
VER:ppast:cli	past participle of verb with clitic
VER:ppre	present participle of verb
WH	wh word

APPENDIX E

This is the table (distributed in electronic format) which students had to fill out with the results of their work. The table were slightly different for each of the four groups, in particular there was a difference in the order of the expressions to research.

Tabella risultati

Ho utilizzato la macchina numero	
Il mio username per CQPWeb è	

Parte 1 (warm-up)			
Espressione inglese	Esempio in inglese	Equivalent e italiano	Esempio in italiano
functional currency	The primary economic environment in which a subsidiary operates determines its functional currency.		

Come sei arrivato alla soluzione (se ci sei arrivato)?

functional currency	
----------------------------	--

In base a quali criteri hai scelto l'esempio italiano?

functional currency	
----------------------------	--

Parte 2			
Espressione inglese	Esempio in inglese	Equivalente italiano	Esempio in italiano
assets given	The cost of the acquisition was measured at the aggregate of the fair values, at the date of exchange, of assets given and liabilities incurred or assumed plus any costs directly attributable to the business combination.		
diluted earnings per share	For diluted earnings per share, the weighted average number of ordinary shares in issue is adjusted to assume conversion of all dilutive potential shares.		
intangible assets	Other intangible assets are stated at cost less accumulated amortisation, which is calculated to write-off their cost, less estimated residual value, on a straight-line basis over their expected useful lives.		

Come sei arrivato alla soluzione (se ci sei arrivato)?

assets given	
diluted earnings per share	
intangible assets	

In base a quali criteri hai scelto l'esempio italiano?

assets given	
diluted earnings per share	
intangible assets	

Parte 3			
Espressione inglese	Esempio in inglese	Equivalente italiano	Esempio in italiano
under	During the year ended 30		

operating leases	September 2010, six aircraft were sold and leased back under operating leases.		
net book value	The net book value of aircraft includes £153.2 million (2009: £148.5 million) relating to advance and option payments for future deliveries of aircraft.		
hedging instrument	Hedge accounting is discontinued when a hedging instrument is derecognised (e.g. through expiry or disposal), or no longer qualifies for hedge accounting.		

Come sei arrivato alla soluzione (se ci sei arrivato)?

under operating leases	
net book value	
hedging instrument	

In base a quali criteri hai scelto l'esempio italiano?

under operating leases	
net book value	
hedging instrument	

APPENDIX F

Questionnaires

Questionnaires 1 and 2 where identical. Computer name and CQPWeb user names were used to match each participant to the queries recorded on the server logs. Some of the general information required was redundant: group membership and system used during the last session could have been inferred from the CQPWeb user name, but it was decided it would be better to have a safety net in case some user made a mistake compiling the questionnaire. This proved to be useful as a couple of users did indeed tick the wrong box sometimes but thanks to the safety measures it was possible to reconstruct the correct information by examining the answer to the other questions and checking the server logs (which contain usernames, IP addresses and queries timestamps). If these extra measures had not been adopted, some users would have had to be excluded and precious data would have been lost.

Questionnaires 1 and 2

1) Appartengo al gruppo:

- A
- B
- C

- D

2) Ho utilizzato il seguente computer:

3) Il mio nome utente per accedere a CQPWeb è:

4) Nella sessione appena terminata ho utilizzato:

5) Nel valutare le affermazioni che seguono:

- immagina che l'utilizzo dei corpora sia essenziale per la traduzione che stai svolgendo (perché i dizionari esistenti non sono sufficienti o perché il committente ti richiede esplicitamente di partire dalla documentazione inserita nel corpus)
- ignora eventuali inconvenienti tecnici che possano essere sorti nel corso della sessione (ad esempio blocco dell'applicazione, funzionalità temporaneamente disabilitate, ecc.)
- lo strumento a cui si fa riferimento è quello che hai utilizzato nella sessione appena conclusa (Carcass o CQPWeb)

Indica quanto sei d'accordo con le seguenti affermazioni:

1. Credo che utilizzerò spesso questo strumento
1 (assolutamente non d'accordo) 2 3 4 5 (assolutamente d'accordo)
2. Ho trovato questo strumento eccessivamente complicato
1 (assolutamente non d'accordo) 2 3 4 5 (assolutamente d'accordo)
3. Ritengo che questo strumento sia facile da utilizzare
1 (assolutamente non d'accordo) 2 3 4 5 (assolutamente d'accordo)
4. Credo che avrei bisogno del supporto di un tecnico per riuscire a usare questo strumento
1 (assolutamente non d'accordo) 2 3 4 5 (assolutamente d'accordo)
5. Trovo che le varie funzionalità dello strumento siano ben integrate
1 (assolutamente non d'accordo) 2 3 4 5 (assolutamente d'accordo)
6. Ritengo che ci siano troppe incoerenze all'interno dello strumento
1 (assolutamente non d'accordo) 2 3 4 5 (assolutamente d'accordo)
7. Credo che la maggior parte degli utenti sia in grado di imparare velocemente a usare questo strumento
1 (assolutamente non d'accordo) 2 3 4 5 (assolutamente d'accordo)
8. Trovo che lo strumento sia molto scomodo da usare
1 (assolutamente non d'accordo) 2 3 4 5 (assolutamente d'accordo)

9. Mi sono sentito molto a mio agio nell'usare questo strumento

1 (assolutamente non d'accordo) 2 3 4 5 (assolutamente d'accordo)

10. Ho dovuto imparare molte cose prima di poter iniziare a usare questo strumento

1 (assolutamente non d'accordo) 2 3 4 5 (assolutamente d'accordo)

6) Ho concluso il questionario e sono pronto a inviarlo (se non sei sicuro delle risposte clicca su "Indietro")

Questionnaire 3

1) Appartengo al gruppo:

- A
- B
- C
- D

2) Ho utilizzato il seguente computer:

3) Il mio nome utente per accedere a CQPWeb è:

4) Indica in quale dei due strumenti:

1. il sort è più facile da usare

1 (Carcass) 2 3 4 5 (CQPWeb)

2. le varie funzioni si trovano più facilmente

1 (Carcass) 2 3 4 5 (CQPWeb)

3. la visualizzazione dei risultati è più chiara

1 (Carcass) 2 3 4 5 (CQPWeb)

4. è più difficile trovare le funzioni

1 (Carcass) 2 3 4 5 (CQPWeb)

5. è più facile passare da un corpus all'altro

1 (Carcass) 2 3 4 5 (CQPWeb)

6. è più facile formulare una query

1 (Carcass) 2 3 4 5 (CQPWeb)

7. la query history è più facile da usare

1 (Carcass) 2 3 4 5 (CQPWeb)

8. si verificano meno imprevisti

1 (Carcass) 2 3 4 5 (CQPWeb)

5) Aggiungi fino a tre altri aspetti per cui secondo te uno dei due strumenti (Carcass o CQPWeb) è migliore dell'altro, spiegando brevemente i motivi della tua preferenza (puoi saltare questa domanda se preferisci)

6) Nel complesso, quale dei due strumenti hai trovato più facile da usare?

- Carcass
- CQPWeb
- Uguali

7) Nel complesso, quale dei due strumenti preferisci? Scegli solo una delle seguenti voci ed indica il motivo della tua scelta nella casella "commenti" qui accanto:

- Carcass
- CQPWeb
- Uguali

8) Per future necessità analoghe di documentazione o traduzione con l'ausilio dei corpora, quale dei due strumenti utilizzeresti? Scegli solo una delle seguenti voci ed indica il motivo della tua scelta nella casella "commenti" qui accanto:

- nessuno dei due
- solamente Carcass
- entrambi, ma preferenzialmente Carcass
- entrambi alla stessa maniera, senza preferenza
- entrambi, ma preferenzialmente CQPWeb
- solamente CQPWeb

9) Quale dei due strumenti consiglieresti di usare a un tuo collega (studente o traduttore) che deve consultare corpora per effettuare una traduzione? Scegli solo una delle seguenti voci ed indica il motivo della tua scelta nella casella "commenti" qui accanto:

- Carcass

- CQPWeb
- Entrambi

10) Anno di nascita (a quattro cifre):

11) Sesso:

- M
- F

12) Lingua madre (è possibile indicarne anche più di una):

13) Indica di seguito tutti i titoli di studio conseguiti a livello universitario (ad es. master, lauree triennali, lauree specialistiche, lauree magistrali ecc.) e la loro denominazione (anche approssimativa, ad es. "laurea triennale in lingue" oppure "laurea magistrale in architettura" ecc.):

14) Prima di frequentare questo corso, avevi già effettuato traduzioni in ambito economico? Scegli una o più delle seguenti voci:

- No
- Sì, nell'ambito di un corso universitario
- Sì, a livello professionale (ossia per conto di un committente)
- Altro:

15) Prima di iniziare questo corso, avevi già utilizzato corpora in altri corsi di traduzione?

- Sì
- No

16) Fai uso di corpora solo quando richiesto esplicitamente da un corso di traduzione oppure anche per altre attività (di traduzione e non)? Scegliere solo una delle seguenti voci

- solo quando richiesto esplicitamente da un corso di traduzione
- anche per altre attività (di traduzione e non)

17) Ho concluso il questionario e sono pronto a inviarlo (se non sei sicuro delle risposte clicca su "Indietro")

APPENDIX G

User comments

These are the comments made by participants in Questionnaire 3 (see 3.1.6.4). Please note that some questions were optional and that not all participants left a comment even when they were invited to. The comments are reported exactly as they appear on the questionnaire, this is the reason why they were not translated into English.

1) Feature comparison

This section shows the answers to the question "Indicate up to three aspects that, in your opinion, make one system better than the other".

User 2

Carcass ha come vantaggi:

- la Smart Search
- i corpora sempre visibili
- il fatto che tutto si svolge nella stessa interfaccia (senza spostarsi di pagina e dover tornare indietro)

Gli svantaggi che ho individuato invece sono: i blocchi frequenti (forse dovuti a troppi utenti che lo utilizzavano allo stesso tempo) e la difficoltà di risalire al testo intero del match.

User 7

Carcass ha la sezione della smart query, che è molto utile perché non bisogna ricordarsi tutta la sintassi CQP ed è più facile tornare alle query già effettuate. L'unica cosa che preferisco in CQPWeb è il modo di visualizzare il contesto, che è più ordinato; in più, il fatto che si può aprire una porzione di testo da sola rende la consultazione meno confusa.

User 10

Con carcass, grazie alla smart query si può evitare di imparare la sintassi CQP

User 11

Ritengo CQPWeb migliore rispetto a Carcass soprattutto per quanto riguarda la visualizzazione del contesto di riferimento per un'occorrenza specifica che intendiamo vedere.

User 13

In Carcass non è possibile (o almeno, non sono riuscito) a ripristinare query utilizzate in passato, quindi ho dovuto digitarle nuovamente. In CQP basta un click per riottenere la stringa nel campo di ricerca. In CQP la visualizzazione del contesto è molto più agevole.

User 15

Per fare una ricerca più dettagliata non è necessario usare LA SINTASSI cqp IN CARCASS

User 27

Su Carcass non sono riuscita a vedere il testo intero e quindi leggere il contesto in modo chiaro.

User 32

Di Carcass trovo molto utile la Smart Query che velocizza il lavoro di ricerca.

User 33

In Carcass devo solo scrivere la parola o espressione che sto cercando, senza specificare altro.

User 36

Secondo me Carcass è migliore di CQPWeb perché la smart query ho molto utile e velocizza le ricerche.

User 37

La funzione per visualizzare il testo completo dei risultati in Carcass non funziona! E quando si espande il contesto, tutti i risultati vengono nuovamente ricaricati, con nuovi tempi di attesa, e spesso c'è il rischio che, durante questa operazione, si blocchi il programma. Questa per me è una cosa molto, molto scomoda. Per questo tenderei a preferire CQPWeb.

User 55

CQPWeb offre una visualizzazione dei risultati più chiara rispetto a Carcass.

User 64

Gli strumenti sono da un punto di vista funzionale pressoché equivalenti, anche se ritengo CQPWeb più scomodo per diversi aspetti (query history, scelta del corpus...)

Naturalmente la Smart Query di Carcass agevola e velocizza di molto la ricerca considerato che la sintassi CQP è di per sé non proprio comoda.

User 80

Con cqp web è più facile il testo completo in cui è inserito il contesto che contiene il lemma cercato però l'interfaccia di carcass è più ordinata, ricorda Antconc.. cioè cqp web è colorato, ma un po' confusionario proprio per il troppo colore.

User 86

L'unico appunto che posso fare a carcass è il fatto che non sempre permette di vedere il contesto per esteso, per il resto l'ho trovato nettamente migliore rispetto a CQPWeb. Il limite di quest'ultimo secondo me sta nel fatto di non permettere di passare facilmente da un corpus all'altro e di tenere sott'occhio entrambi a meno che non si aprano più schede. Sicuramente questi aspetti sono meno pesanti con la possibilità di consultare un dizionario.

User 89

Non ho capito come cavolo si apre il contesto su uno dei risultati di carcass non ho usato il sort... e al momento manco ricordo a cosa serve sono OT, vero-

Carcass migliore per l'interfaccia, perché ti fa vedere tutte le finestre, mentre cqpweb non riesce a visualizzarle contemporaneamente

2) Ease of use

This section collects the reasons given by users for their answer to the question "which of the two system did you find easier to use".

Users who preferred Carcass**User 4**

lo trovo più intuitivo

User 7

È più chiaro e la Smart query è utilissima

User 10

perchè c'è la smart query e mi sembra tutto più chiaro

User 12

L'ho trovato leggermente più pratico e funzionale

User 30

nel complesso preferisco la struttura di Carcass

User 32

È sostanzialmente una preferenza dovuta all'interfaccia del programma. Carcass mi sembra più chiaro e ho l'impressione di trovare i comandi più facilmente. L'unico problema di carcass è che i risultati li dà in fondo alla pagina che bisogna sempre allargare per vedere i risultati più chiaramente, ma non è certamente un problema fondamentale.

User 33

Più facile da utilizzare, quindi perdo meno tempo nella ricerca.

User 35

Con la smart query è più semplice effettuare delle ricerche.

User 36

E' più veloce l'interrogazione

User 40

mi sembra più semplice e funzionale come strumento.

User 41

In entrambi ci sono una simple a una expert query, ma la Smart query di Carcass facilita e velocizza molto la ricerca.

User 52

Nonostante io trovi molto faticoso lavorare solo con i corpus mi sono trovata molto meglio con Carcass perchè le ricerche a mio parere sono più semplici da effettuare e mi sembra che l'interfaccia si possa gestire più rapidamente e in modo più pratico. Mi sembra meno complesso

User 58

mi sembra più chiaro e immediato

User 64

Più agile da usare

User 83

Per quanto riguarda Carcass, secondo me è rivedibile la modalità di allargamento del contesto, nel senso che sarebbe più utile una visualizzazione come quella di CQPWeb. Per il resto non mi è dispiaciuto. In compenso CQPWeb è poco pratico per saltare rapidamente da un corpus all'altro e non possiede una modalità di ricerca "semplificata" come la smart query.

User 84

Trovo che la funzione smart query faccia risparmiare molto tempo.

User 91

Più semplice e funzionale.

Users who preferred CQPWeb**User 3**

La visualizzazioni dei risultati con Carcass non è molto chiara.

User 11

Risulta migliore la visualizzazione del contesto di riferimento per le occorrenze trovate.

User 13

All'apparenza più complicato, ma più comoda in fine dei conti

User 27

più chiaro e pratico

User 31

Più facile prendere visione del contesto

User 51

ha un'interfaccia più semplice ed è più facile da usare

User 53

è più semplice formulare query e la visualizzazione appare più chiara rispetto all'altro programma

User 55

È molto più semplice e comodo da utilizzare.

User 59

L'ho trovato più intuitivo e pratico da utilizzare.

User 60

L'ho trovato un po' più facile da usare anche se passare da un corpus all'altro è meno immediato

User 76

Per qualche strana ragione ho trovato più facilmente i risultati usando CQPweb e nonostante non abbia un'interfaccia ma sia più basico sembra che funzioni meglio, almeno per le ricerche di questo esperimento.

User 81

CQP web è più veloce nel fornire risultati ed è più facile consultarli.

User 85

I risultati mi sembrano più facili da leggere e il sort funziona molto meglio.

User 89

carcass torna più utile solo sulle ricerche molto, ma molto complesse

User 90

più semplice e immediato

Users who liked them both equally**User 14**

Ci sono alcuni aspetti di CQPweb che preferisco (per esempio, si vedono meglio i risultati delle query) e altri che preferisco di Carcass (per esempio, è più semplice passare da un corpus all'altro).

User 77

Tutti due hanno degli svantaggi e aspetti positivi. Non potrei scegliere tra i due.

3) Would use again

These are the reasons given by users for their answer to the question "if in the future you had to perform a similar translation and/or documentation task, which of the two systems would you use?"

Users who would not recommend either system**User 33**

Conosco altri strumenti con cui mi trovo meglio.

Users who would recommend both systems**User 36**

L'unico svantaggio di Carcass è che deve essere installato

Users who would recommend CQPWeb**User 14**

I risultati delle query sono più chiari e la funzione per espandere il contesto funziona meglio rispetto a CQPweb.

User 31

mi piace di più

User 51

CQPWeb è più semplice da usare

User 55

E' più semplice da utilizzare e i risultati sono visualizzati meglio.

User 60

Trovo che CQPWeb sia leggermente più facile da usare

User 85

Mi sembra più facile nel complesso

Users who would recommend Carcass**User 7**

Carcass mi sembra più semplice da usare, ma il modo in cui si visualizza il contesto mi risulta confuso.

User 9

con un po' più di pratica anche CQPWeb è un ottimo strumento

User 10

mi sembra più intuitivo carcass

User 28

userei CQPWeb per comprendere meglio il contesto

User 32

Per le ragioni che ho espresso nella pagina precedente, consiglierei Carcass solo per una questione di preferenza di interfaccia e per la possibilità di usufruire della Smart Query.

User 35

Carcass mi sembra più intuitivo da utilizzare.

User 52

Utilizzerei solo Carcass perchè credo mi possa permettere di fare tutto quello che faccio con CQPweb ma mi risulta più comodo usarlo

User 58

trovo che sia più chiaro, ma allo stesso tempo non rinuncerei alla possibilità di usare CQPWeb

User 64

Per la funzione Smart Query

User 77

Si cambia più facilmente tra i corpus

4) User recommendation

This section collects the reasons given by users for their answer to the question "which of the two systems would you recommend to a colleague (student or translator) who had to use corpora for a translation?".

Users who would recommend Carcass

User 33

Più facile da usare.

User 38

E' più chiaro

User 76

è più facile da usare a prima vista

Users who would recommend CQPWeb

User 36

Consiglierei CQPWeb perché non deve essere installato alcun programma

User 55

E' più semplice da utilizzare e i risultati sono visualizzati meglio.

Users who would recommend both systems

User 7

Carcass ha il vantaggio di poter essere usato anche senza conoscere bene la sintassi CQP, ma non mi piace il modo in cui si visualizzano i contesti.

User 10

sono entrambi buoni per consultare i corpora. magari cqpweb è di più facile accesso perchè è online.

User 13

Spiegandogli i pregi e difetti di entrambi

User 14

Li trovo entrambi funzionali per il lavoro del traduttore.

User 32

Tutto sommato, sono due strumenti molto simili e che interrogano gli stessi corpora quindi li consiglierei in maniera tutto sommato indifferente.

User 35

Permettono di confrontare dei corpora utilizzando diverse funzioni.

User 51

Sono entrambi validi

User 58

Perchè una volta imparato il linguaggio tecnico, è vantaggioso utilizzarli entrambi

User 59

dipende dal tipo di ricerca da effettuare e anche dal traduttore stesso da con quale si trova più a suo agio

User 60

Entrambi perché anche Carcass ha dei lati positivi se si impara bene ad usare la sintassi

User 64

Nonostante preferisca Carcass credo che gli strumenti si equivalgano.

User 77

Molto utile per trovare il contestto.

User 81

Si può facilmente imparare ad utilizzare entrambi gli strumenti.

User 85

sono entrambi utili ma credo sia importante imparare bene ad usarli.

APPENDIX H

Tutorials

The two tutorials reproduced here have been created to offer a quick introduction to Carcass, they are available online in the "Help" section of the project's official web page: <http://sarcophagus.sslmit.unibo.it/>. The online versions will be updated as new versions of the software are released, therefore they might differ from the one reproduced here.

Carcass Basic Tutorial

Welcome to the Carcass Basic Tutorial!

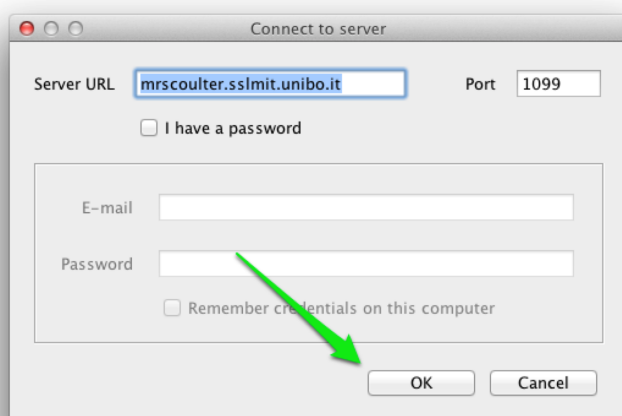
Introduction

This tutorial assumes no prior knowledge of the tool, and will walk you through the process of connecting to a remote server, submitting your first query and browsing concordance lines.

Carcass is a corpus query tool. It is a client application, which means it needs a working Internet connection to log into a server. For now, the only corpora you can use with Carcass are those hosted on the server (i.e. at the moment it is not possible for regular users to add corpora to the system).

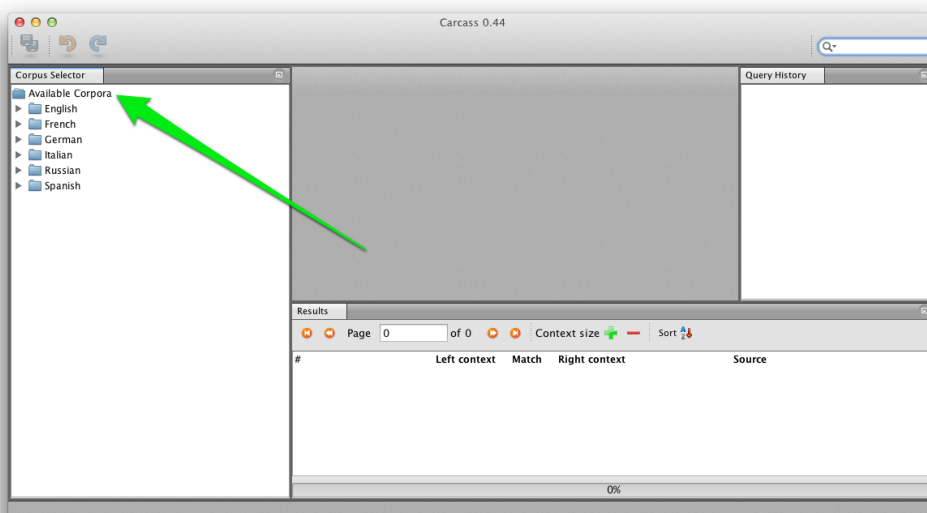
Connection to the server

Immediately after starting the program, a dialog window will appear, requesting the parameters for the connection to the server. You don't need to do anything here, just accept the default values by clicking on the "OK" button.

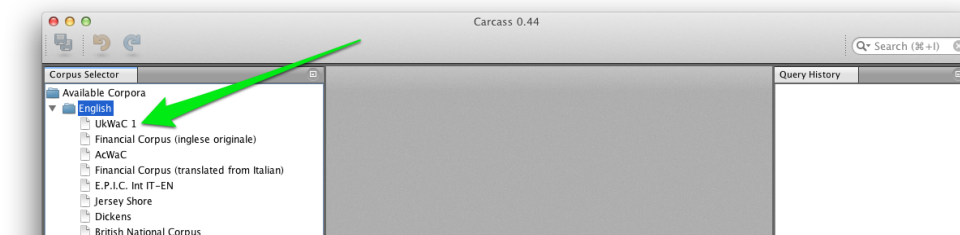


Selecting a corpus

After a few moments the main window of Carcass will appear, on the left you'll see a list of all available corpora, grouped by language.

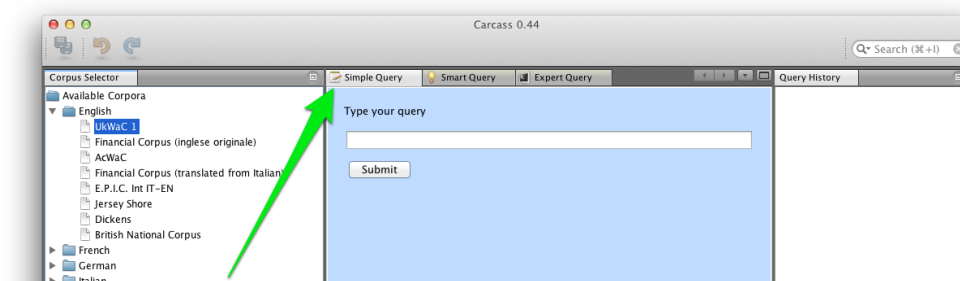


Click on "English" to display all available English corpora.



Click on "UkWaC 1" to select the UkWaC corpus (more information on this corpus are available on the Wacky project's home page).¹

As soon as you select the corpus, the query editors will appear:



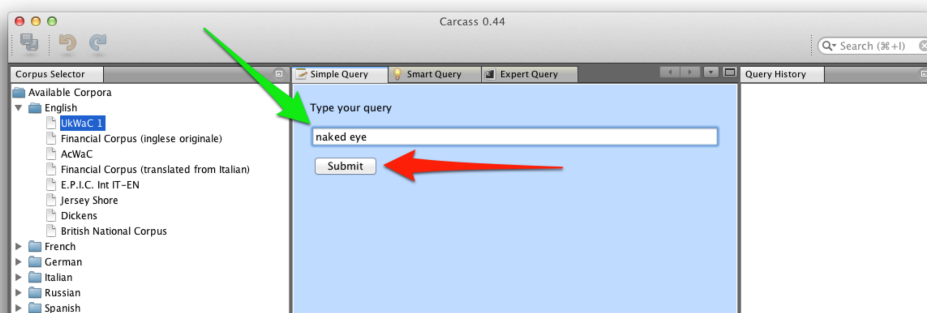
As you can see from the picture above, you can choose between Simple, Smart and Expert query editor.

Submitting a query

For now let's just stick to the Simple editor (the Smart and Expert editors are covered in the advanced tutorial). Type the following phrase in the box and click on Submit:

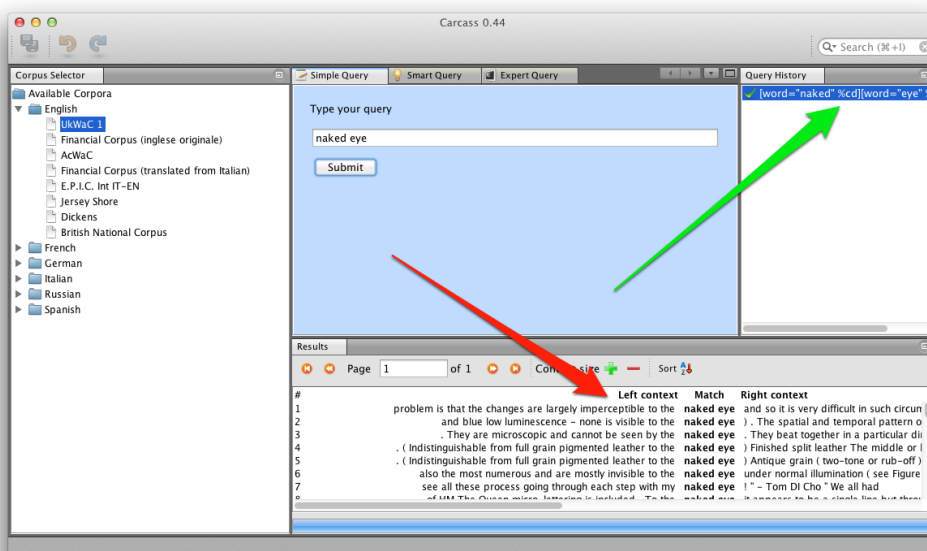
naked eye

¹ <http://wacky.sslmit.unibo.it/>



Browsing results

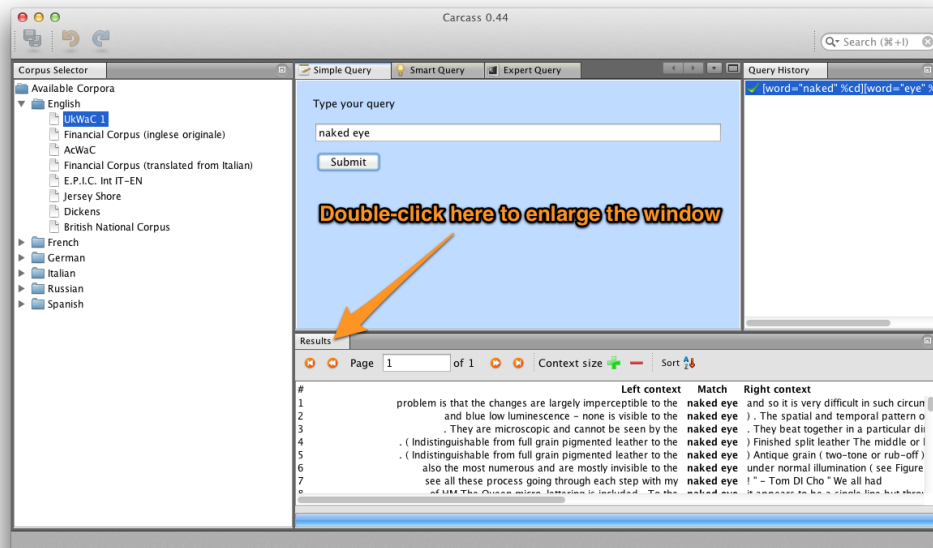
After a few seconds the results of your query will appear:



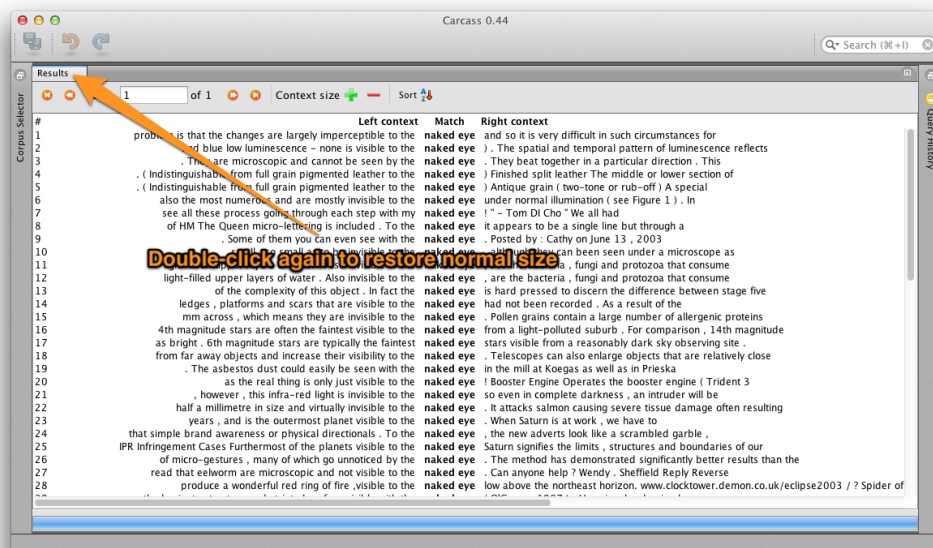
In the *Results panel* (marked by the red arrow) you'll see the concordance lines retrieved from the corpus.

The query you submitted is displayed in the *Query History* (marked by the green arrow), all the queries you submit will appear in this panel. You'll notice that the syntax used in the *Query History* appears to be more complex than the one you used for your *Simple query*, we'll cover this in the advanced tutorial (see below).

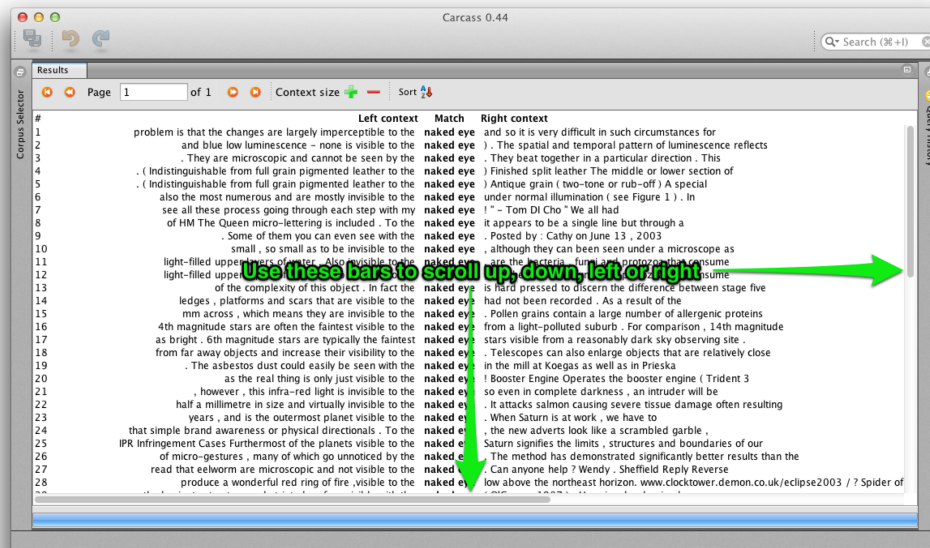
It is possible to enlarge the *Results* area by double-clicking on the tab:



To restore the window to its normal size, double-click on the tab again:



You can use the scrollbars to move up, down, left or right:

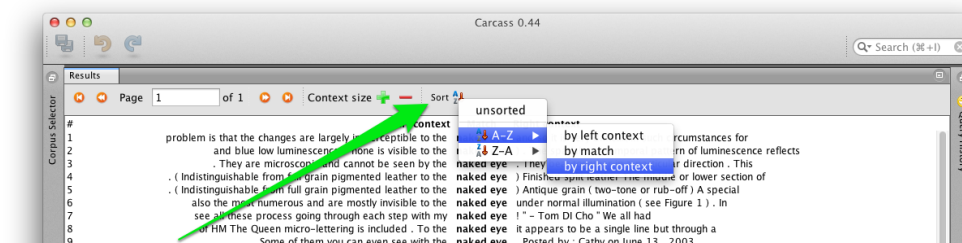


Sorting results

We will now sort the results: click on *Sort* and then select *A-Z* → *by right context*. The concordance lines will be sorted by right context: you'll notice that punctuation marks (exclamation points, parentheses, etc.) appear first in the sort order.

You can also sort by **left context** or by **match** (this is only useful if you are not looking for an exact phrase like we did in this example). If you choose *Z-A* instead of *A-Z* results will be sorted in **inverted alphabetical order**.

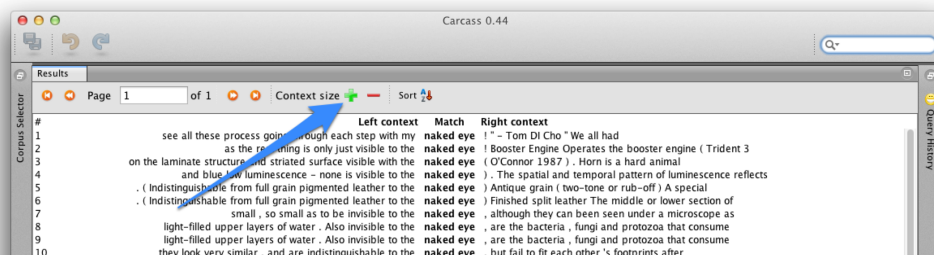
Finally **unsorted** restores the original order of the concordance lines (i.e. the order in which they appear in the corpus).



Context size

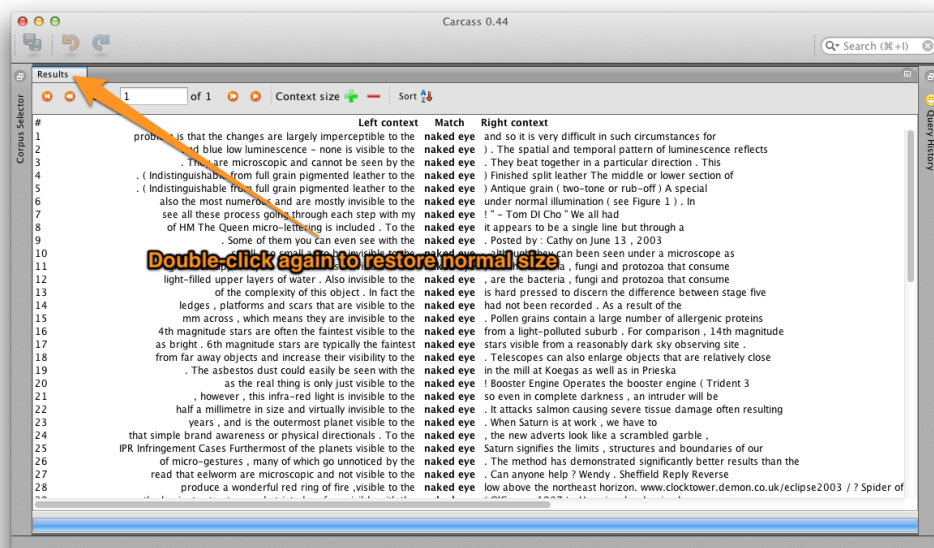
Carcass by default displays 10 tokens on either side of the match, it is possible to increase the context size by clicking on the "+" button in the *Context size* section of the toolbar.

The context increases to 20, 50 and 100 tokens on both sides every time the "+" button is clicked. It decreases by an equal amount every time the "-" button is clicked (to a minimum of 10 tokens).



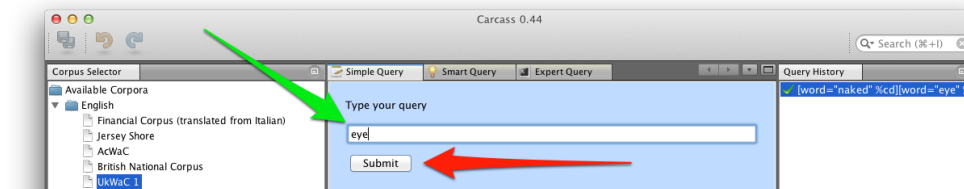
Query history

First of all, if you haven't done it already, restore the results window to its original size:

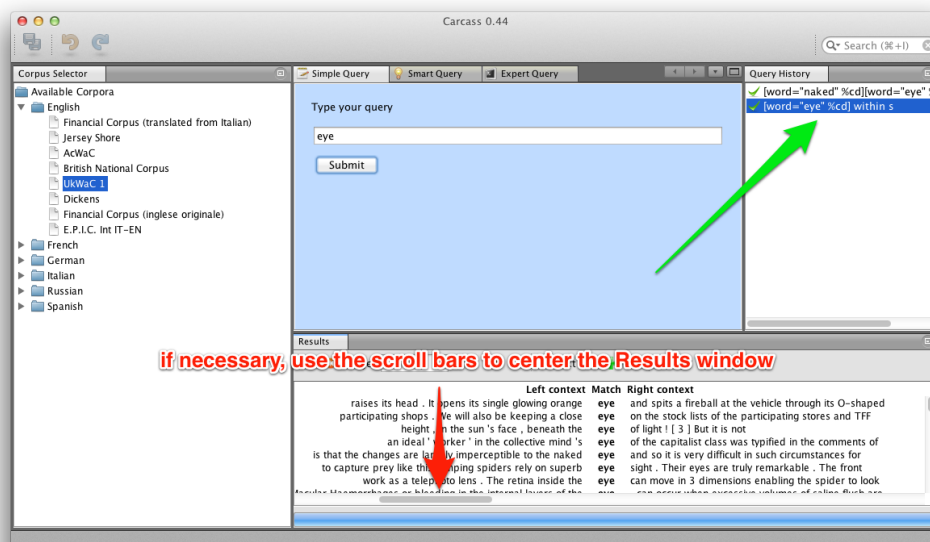


Then, type a new query in the box, this time it's a single word:

eye

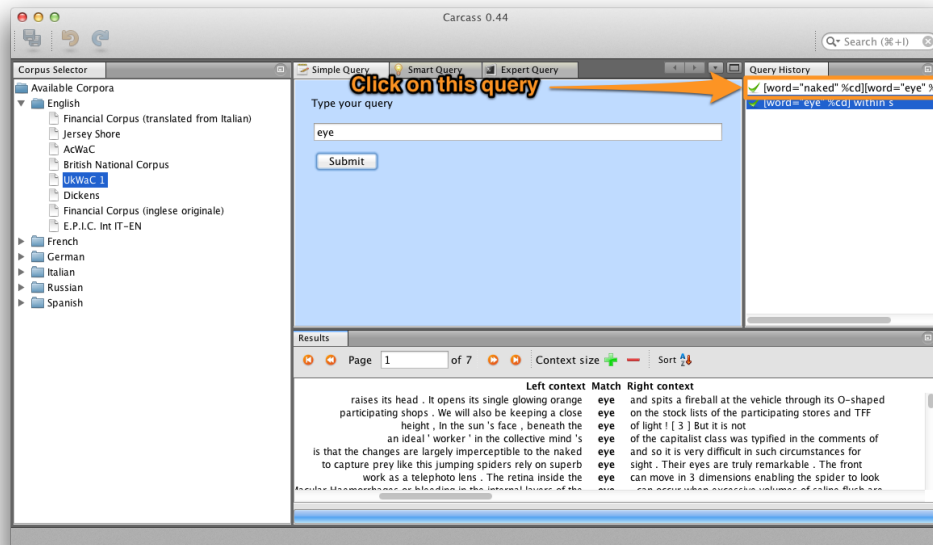


After a few seconds the results will appear (you may need to use the scrollbar to center the Results window and see the first concordance lines):



You can browse the results, sort them and increase/decrease the context size like you did for the previous query. Remember that you can also enlarge the Results window by double-clicking on the tab.

You'll also notice that the new query was added to the Query history on the right side of the application. Try clicking on the previous query in the history (the one starting with `[word="naked" %cd]`):

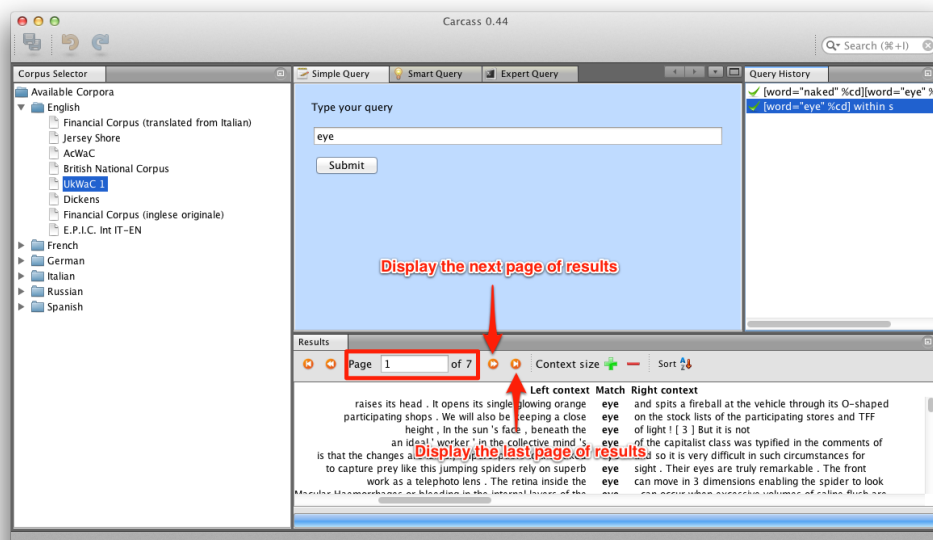


The concordance lines from the previous query will be displayed in the Results panel. You can switch back and forth between all the queries in your history as you like.

Result pages

Carcass displays a maximum of 1.000 concordance lines at a time, if the query returns less than 1.000 hits (as is the case for the query "naked eye") you don't need to worry, but in the case of the query for "eye" we get more than 6.000 hits.

You'll notice that the number of pages of results is displayed at the top of the result window:

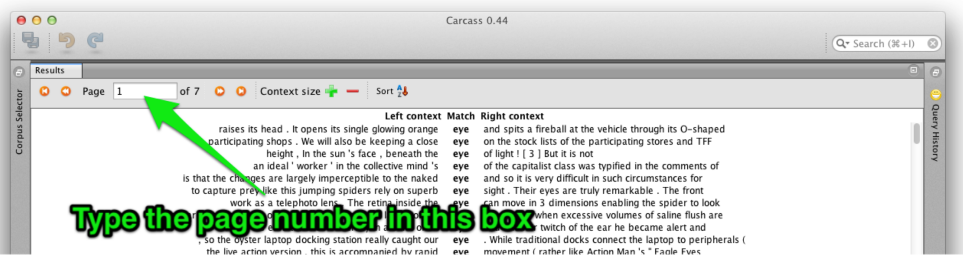


In this case we have a total of 7 pages (the first one displays results from 1 to 1000, the second displays results from 1001 to 2000 and so on). You can click on the Next Page button to display to the next page of results, or to the Last Page button to display the last. Likewise, the Previous Page and First Page buttons display the previous and first page of results respectively.

You can also jump directly to any page by typing its number directly in the box:

- type **4** in the page number box
- press Enter on your keyboard

Page 4 will be displayed directly.

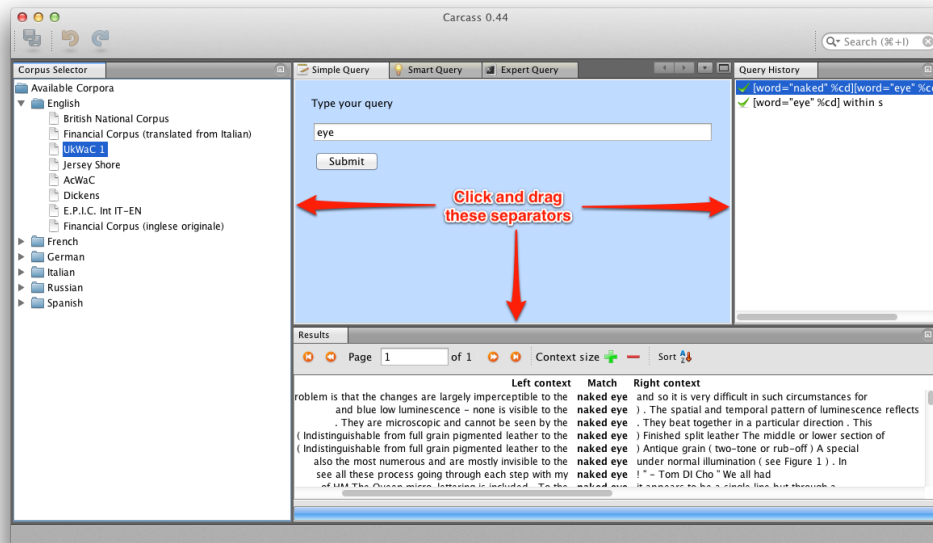


Resizing panels

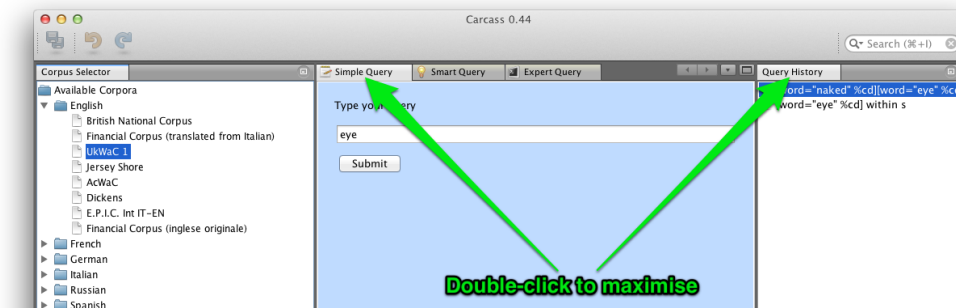
Carcass allows you to resize various parts of the application to suit your needs.

Try the following:

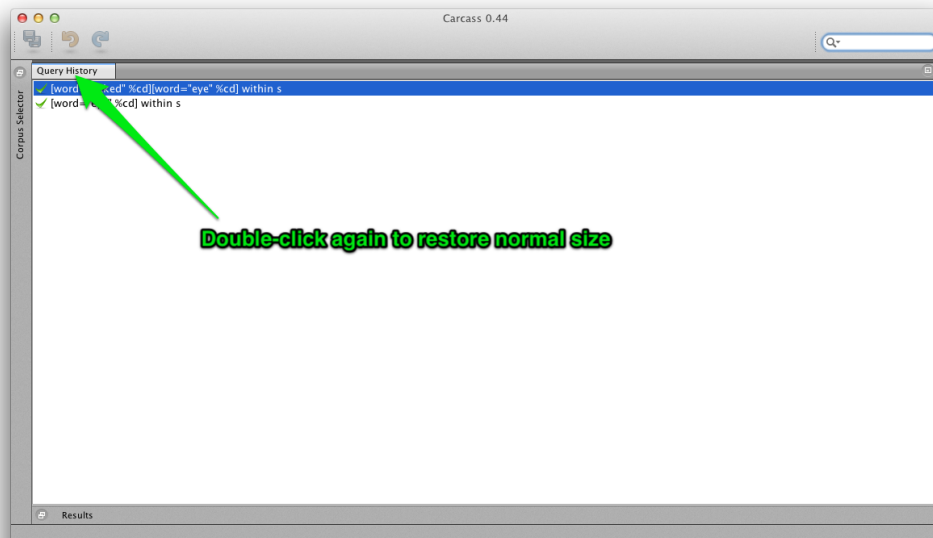
1. move the mouse exactly over the line that separates the *Simple query* panel (the blue one) from the *Results* panel right below it, you'll notice that the mouse pointer changes
2. without moving the pointer, click with your mouse and, while keeping the button down, move the mouse up or down, you'll see that the sizes of the two panels change.



We already saw that it's possible to maximise the Results window by double-clicking on the tab title, you can do the same with the Simple query editor and with the Query history:



To restore the normal window size, double-click on the tab title:



Maximising windows is especially useful when queries are particularly long and don't fit in the regular window.

Conclusion

This concludes the basic tutorial, if you want to learn about the more advanced features of Carcass, check out the advanced tutorial.

Carcass Advanced Tutorial

This tutorial is based on Carcass version 0.44.

Welcome to the Carcass Advanced tutorial!

Prerequisites

This guide assumes you already completed the basic tutorial. If you haven't done it already, we recommend you complete it and then come back here.

Introduction

To begin this tutorial, follow these steps:

- start Carcass and connect to the server
- in the *Corpus Selector* click on "English"
- click on UkWaC 1

Using the asterisk

Let's say you want to look for words that begin with "under" (such as "under", "underneath", "understand", etc.)

You can easily do this in Carcass using the asterisk.²

Type this query in the Simple Query box and click on Submit:

under*

² The correct term for this is actually "Kleene star operator", but we'll just call it asterisk to keep things simple.

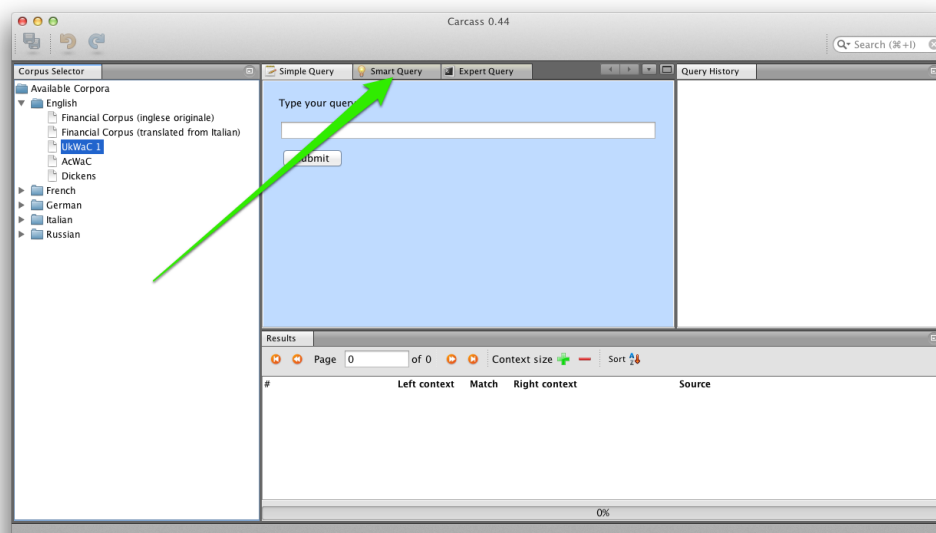
The query will return results like "under", "underpinning", "underside" etc.

N.B.: this simplified use of the asterisk only applies to the *Simple Query Editor*, the *Smart* and *Expert Query Editors* use a slightly different syntax. This may change in the future.

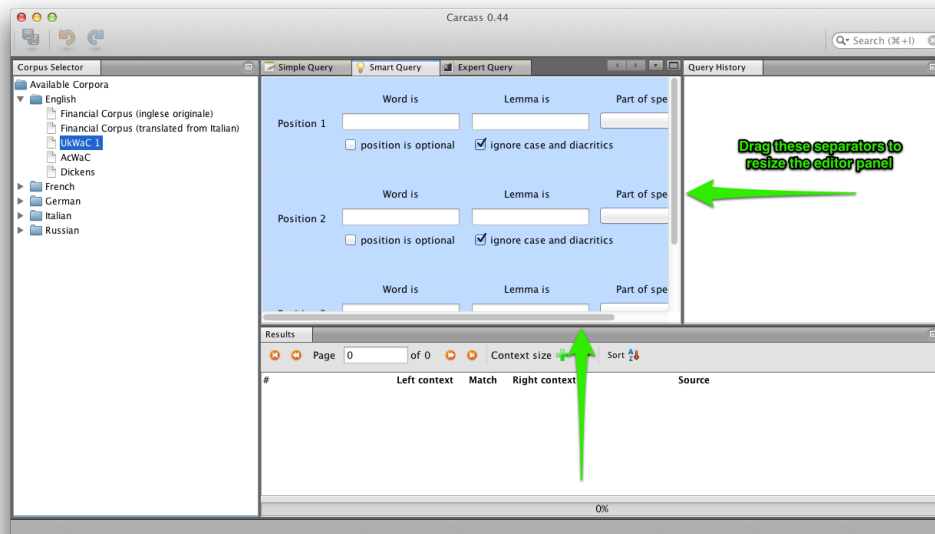
The Smart Query Editor

In this section you'll learn how to use the *Smart Query Editor*, a special editor that allows you to compose queries which exploit corpus annotation (things like part-of-speech tags and lemma).

Select the Smart Query Editor.

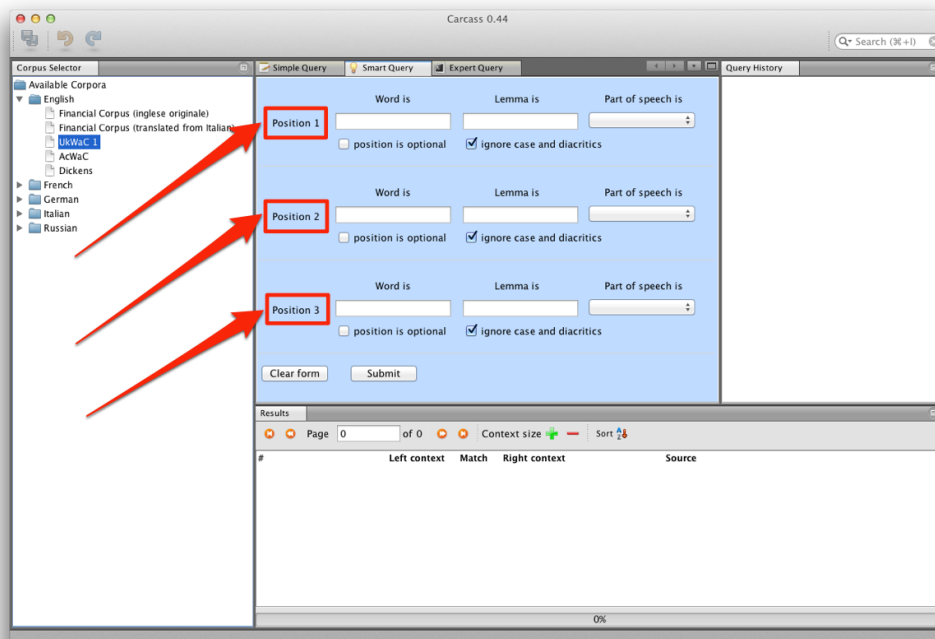


The Smart query editor will open. If your computer has a small screen it may look like this:



You may want to maximise the application window at this point. Remember also that you can drag the separators to redistribute the space in the application (see the basic tutorial for more info on this).

The application should look more or less like this at this point (you should be able to see all the elements in the *Smart Query Editor*):



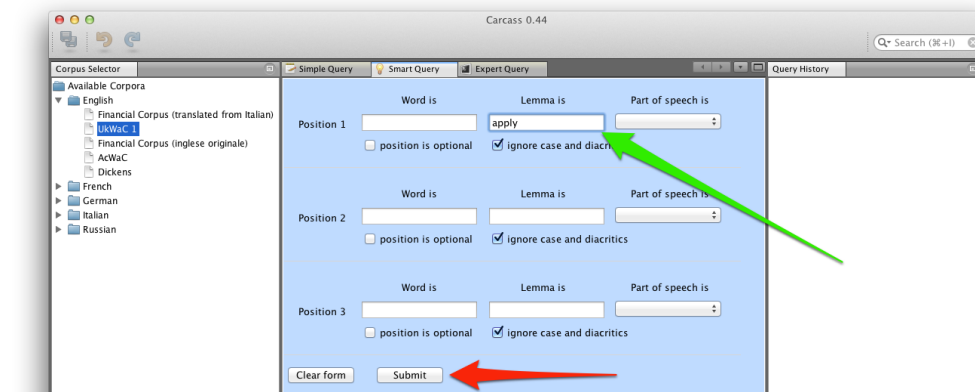
The smart editor contains three *Position descriptors*, each of these represents the constraint you want to put on a word in your query. You don't need to use all three positions in your query, nor do you need to use all the options offered by the editor (more on this later).

For instance, let's say that you want to look for examples of use of the verb "apply" (in all its inflected forms) in the UkWaC corpus.

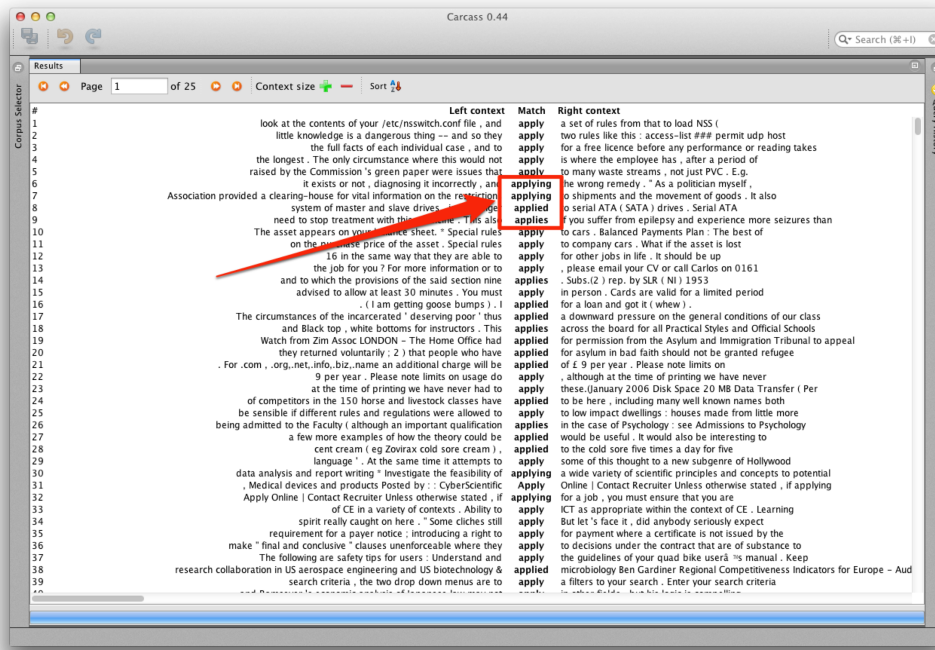
In the *Lemma* box of Position 1 type:

apply

Leave everything else untouched and click on *Submit*.



In a few seconds the results will appear and you'll notice that they also include inflected forms of the verb "apply" (i.e. "applying", "applied", "applies", etc.).

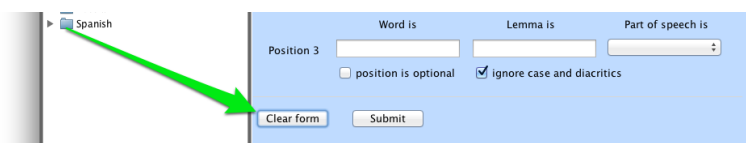


More on the Smart Query Editor

Note also that the query you just submitted was added to the Query history, just like the simple query you submitted using the Simple Query Editor.

Now let's try a different query:

- first of all, reset the previous query by clicking on Clear form



- then fill out the form like this:
 - Position 1: in the *Lemma* box type apply
 - Position 2: in the *Part of speech* drop-down list select Determiner
 - Position 3: in the *Part of speech* drop-down list select Noun
 - click on *Submit*

The screenshot shows a query builder interface with three positions. Position 1 has 'Word is' set to 'apply'. Position 2 has 'Part of speech is' set to 'Determiner'. Position 3 has 'Part of speech is' set to 'Noun'. A red arrow points to the 'Submit' button.

This query returns all occurrences of the the verb "apply" (in all its inflected forms), followed by a determiner and a noun.

The screenshot shows the search results page. The matches are: 'apply a set', 'apply the guidelines', 'apply a filters', 'apply the policies', 'apply these concepts', 'apply the dolphin', 'applies a combination', 'apply the name', 'apply the truth', 'apply the decision', and 'Apply The Postgraduate'.

There might be errors in the results (e.g. sometimes the second word will not be a determiner or the third will not be a noun): they are due to inaccuracies in the pos-tagging of the corpus. This kind of errors are to be expected when dealing with automatically tagged corpora (such as the UkWaC corpus we are using in this tutorial).

Here again, sorting results by match could be useful (see the basic tutorial if you don't remember how to do it).

Optional positions

In the smart query, it is also possible to specify that a particular position is optional.

For example, we could try to look for the same pattern as before ("apply" followed by a determiner and then by a noun) and specify that we want the determiner to be optional.

We will modify the query we just submitted by ticking the position is optional checkbox in Position 2. If you didn't modify anything after submitting the last query, all you have to do is tick the checkbox marked by the red arrow:

This query will return concordances like "apply makeup" and "apply the policies" because we told the program that position 2 was optional.

If we were interested in finding out what you can do with a "proposal" in English, we could try a query like this

- click on *Clear form* to reset the form
- Position 1: in the *Part of speech* drop-down menu select Verb
- Position 2: tick the *position is optional* checkbox and then select Determiner in the *Part of speech* drop-down menu
- Position 3: in the *Lemma* box type proposal

N.B.: this query will probably take more than a few seconds to complete.

The query will return concordances like "made proposals", "reject the proposal", etc.

Using multiple constraints

It is also possible to use multiple constraints on a single position. If we wanted to look for verbs starting with "under" followed by an optional determiner and then a noun, we could compose a query like this:

- click on *Clear form*
- Position 1: in the *Word* box type "under.*" (without quotes, note the dot before the asterisk), in the *Part of speech* drop-down menu select Verb
- Position 2: in the *Part of speech* drop-down menu select Determiner, tick the *Position is optional* checkbox
- Position 3: in the *Part of speech* drop-down menu select Noun
- click on *Submit*

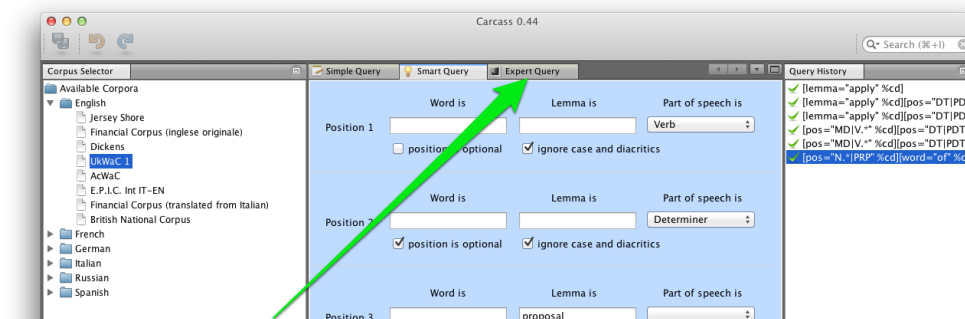
The screenshot shows the 'Expert Query' tab in the software. On the left, the 'Corpus Selector' lists various corpora including English, E.P.I.C. Int (IT-EN), British National Corpus, ACWaC, Financial Corpus (inglese originale), Dickens, Jersey Shore, and Financial Corpus (translated from Italian). The main area is divided into three sections for 'Position 1', 'Position 2', and 'Position 3'. Each section has fields for 'Word is', 'Lemma is', and 'Part of speech is'. Position 1 has 'under.*' in the Word field and 'Verb' in the Part of speech dropdown. Position 2 has 'Determiner' in the Part of speech dropdown and the 'position is optional' checkbox is checked. Position 3 has 'Noun' in the Part of speech dropdown. There are also checkboxes for 'ignore case and diacritics' in each position. At the bottom are 'Clear form' and 'Submit' buttons. On the right, the 'Query History' shows a list of queries with their corresponding CQP syntax.

The query will return concordances such as "undertaken change" and "understand the complexities". You should note that the syntax we used here for the asterisk is different from the one we used in the first example of the tutorial: here we used `.*` instead of a simple asterisk to represent any sequence of characters. This inconsistency will probably be resolved in future versions of the software.

Expert Query Editor

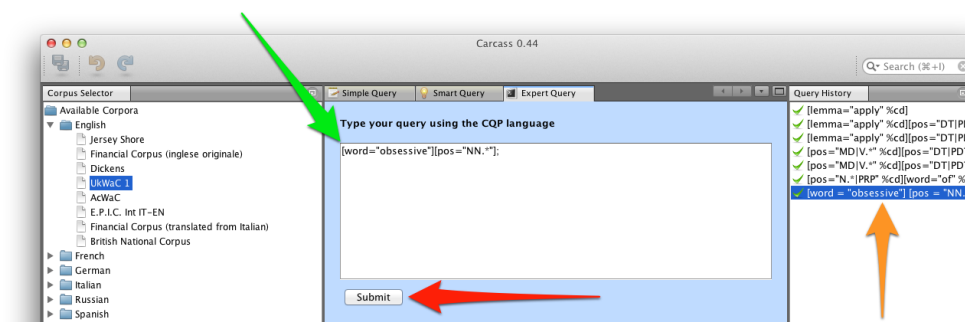
The most advanced query editor available in Carcass is the *Expert Query Editor* which allows you to use the full CQP syntax.

To open the *Expert Query Editor* click on the tab marked by the green arrow in the picture below:



The Expert Query Editor consists of a single text box where you can type your query directly using the CQP language. Type the following query and then click on *Submit*:

`[word="obsessive"] [pos="NN.*"];`



The query will take a few seconds to complete: it will appear in the query history as usual and the results will be displayed at the bottom of the screen.

Now you can browse the results or sort them just like you did with the results from the *Simple* and *Smart* query editors.

The CQP language used by the Expert Query Editor is rather complex and is beyond the scope of this tutorial, for more information refer to the official CQP tutorial.³

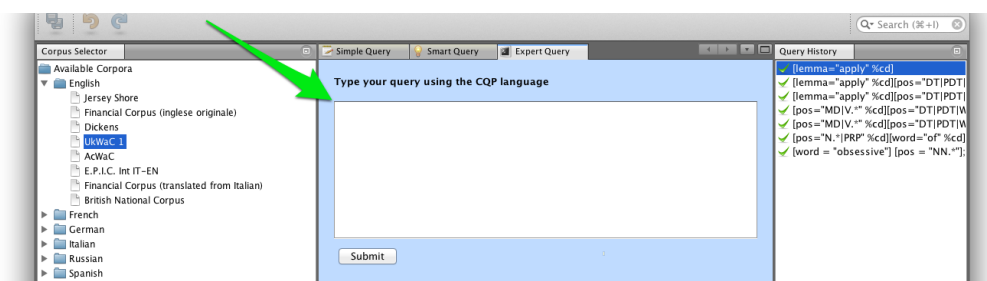
3 <http://goo.gl/9oyVC>

Query History

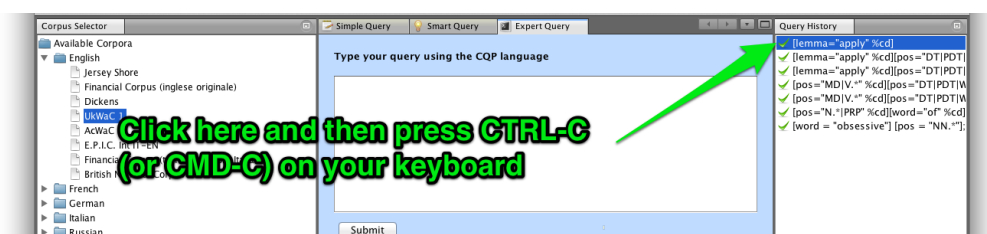
You may have realised by now that the queries listed in the *Query History* are all expressed using the CQP language: if you use the *Simple* or the *Smart* query editors, Carcass automatically "translates" your queries into the CQP language.

We've seen how you can recall the results of previous queries by clicking on them, but there is another interesting thing you can do:

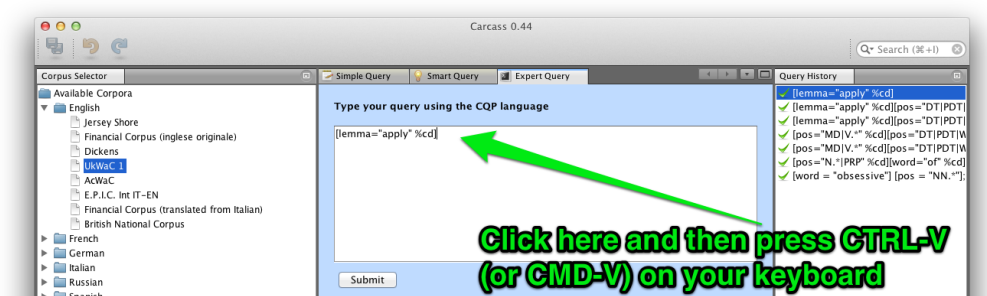
- delete the content of the Expert Query Editor box, which should look like this



- in the Query history, click on the first query you submitted



- press **CTRL-C** (or **CMD-C** if you are using a Mac) on your keyboard, this will copy the query to the clipboard
- click in the empty text box in the Expert Query Editor and then press **CTRL-V** (or **CMD-V** if you are using a Mac) on your keyboard
- the CQP query will be pasted in the box



This is particularly useful if you want to start composing a query in the Smart Editor and then modify it manually.

Conclusion

This concludes the Carcass Advanced Tutorial.